ADDRESSING COMPLEXITY IN ENTERPRISE ARCHITECTURE

LOUKAS KONSTANTINOU

This dissertation was submitted in part fulfilment of requirements for the degree of MSc Enterprise Information Systems

DEPT. OF COMPUTER AND INFORMATION SCIENCES

UNIVERSITY OF STRATHCLYDE

SEPTEMBER 2015

Declaration

This dissertation is submitted in part fulfilment of the requirements for the degree of MSc of the University of Strathclyde.

I declare that this dissertation embodies the results of my own work and that it has been composed by myself. Following normal academic conventions, I have made due acknowledgement to the work of others.

I declare that I have sought, and received, ethics approval via the Departmental Ethics Committee as appropriate to my research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to provide copies of the dissertation, at cost, to those who may in the future request a copy of the dissertation for private study or research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to place a copy of the dissertation in a publicly available archive.

(please tick) Yes [ ] No [ ]

I declare that the word count for this dissertation (excluding title page, declaration, abstract, acknowledgements, table of contents, list of illustrations, references and appendices) is

I confirm that I wish this to be assessed as a Type 1 2 3 4 5 Dissertation (please circle)

Signature:

Date:

Abstract

Information Technology (IT) rapidly becomes a key factor in the competitiveness and sustainability of most enterprises; however, moving away from the support role which previously held, IT architecture inherits the complexity of the business itself. In order to be successful in this new function, IT should be redefined holistically as Enterprise Architecture (EA), aiming to support and simplify every aspect of business functionality.

The aim of this dissertation is to define EA complexity and its root causes and discuss possible methods and techniques of tackling the issue. A number of complexity factors and complexity management suggestions derived from literature are reassessed through involvement in the analysis and design stages of an actual enterprise-level application development.

Findings revealed that a substantial majority of these factors were indeed evident in practice; moreover, a deeper understanding of EA complexity management is developed, as the set of theoretical guidelines is refined in the light of the specific scenario. These guidelines can be roughly categorised in proposed analysis/design techniques (appropriate scoping, partitioning and modelling of the architecture), suggestions related to application of standards and architectural artefact management, as well as recommendations related to stakeholder management and requirements engineering.

Acknowledgements

This dissertation project has been a very rewarding experience, marking the completion of my postgraduate studies in the University of Strathclyde. However, it would have been very difficult to be realised without the help of certain people. Therefore, I would like to thank my supervisor for his valuable guidance and feedback, as well as everyone at the Development & Innovation team of the Information Services, especially Donna Brawley and Emma Graham, for allowing me the chance to carry out the case study component in such a supporting, helpful and friendly environment.

Glasgow, August 2015

Loukas Konstantinou

Table of Contents

List of Illustrations

1. Introduction

This study investigates the open issue of managing complexity in the field of Enterprise Architecture. As organizations continuously aspire to benefit from rapid technological progress, they have found themselves increasingly dependent on Information Technology (IT); the role of IT in enterprises has shifted from merely supporting and facilitating business towards producing key competencies, market opportunities and competitive advantage. Naturally so, IT is also treated differently; regarded as a strategic tool, IT management and planning takes a holistic viewpoint in the form of Enterprise Architecture (EA). Large enterprises cannot deploy effective Information Systems without an overarching, high-level plan which would enable them to confront the presence of numerous disparate systems, multiple business environments in different sectors or geographic regions; the complexity embedded is far too great to be addressed by ad hoc, unit-level decisions and solutions. The issue of addressing complexity is the single greatest challenge that EA has to face; otherwise, it is of little business value.

Given that EA as a discipline is relatively new, literature on proposed strategies is limited when compared to more mature engineering fields. When it comes to suggestions on how to tackle the problem of decoding complexity, sources are even scarcer. Commonly adopted EA frameworks (e.g. TOGAF, Zachman, FEAF) are deliberately lacking detailed, comprehensive guidelines on how to carry out the proposed methodologies, in order to be as universal as possible. Consequently, the purpose of this dissertation project is to gain theoretical and practical understanding of managing system complexity and possibly contribute to the EA discipline, based on the in-practice techniques investigated in the case study.

The research questions on which this project is based are as follows; what are the factors causing EA to be complex? Subsequently, what tools and techniques can be utilised in order to address EA complexity? Lastly, this project aspires to explore and reflect on how complexity is managed in practice, as well as the potential challenges that might arise, based both on literature findings and on the experience of involvement in part of the Mobile Application project at the Information Services department of the University of

Strathclyde. The present research is mainly focused on two particular architectural domains (i.e. Business and Data), specifically within the analysis and design stages of EA development; these decisions were made partly due to Business and Data subsets being key complexity factors and partly in order to maintain a realistic scope for the project within the available time period.

The project will follow the structure presented below; chapter 2 describes the strategy and methodology used for this research. Chapter 3 is focused on the literature review, which aims to define EA complexity and present a set of existing propositions on how it should be addressed. Chapter 4 concentrates on the case study, the project of handling university events management at an institution level, serving to test hypotheses derived from literature as well as identifying practices not previously studied; lastly, the concluding chapter is focused on the analysis of the findings from the fieldwork, discussing the applicability of the various principles and identifying opportunities for further research on the matter.

2. Methodology

This chapter will discuss the strategy and methodology of the research process leading to this project. Universally accepted research methods, derived from relevant bibliographic sources, are briefly described in order to justify which of them were selected for the dissertation. The chapter is structured as follows; first, the nature of the research is defined and explained; next, an overview of the research strategy is provided; lastly, the framework and selected methods for each of the dissertation components are thoroughly discussed.

2.1 Nature of Research

Research can be classified into two major categories; qualitative and quantitative studies. Depending on the purpose and the expected outcomes of the project at hand, one of the two might be deemed more suitable; there is no point in debating one over the other, as each is focused on different types of research and usually related to different research disciplines.

Quantitative research, usually applicable in natural and practical sciences, is focused towards studying isolated occurrences in order to test related previous theory. In order to achieve that, research includes precise specification of the testing variables, whose values are measured in a countable manner and are often prone to statistical analysis (Creswell, 2013). This sort of studies is usually carried out by means of an experiment-based research, whose outcome is used to test prior hypotheses and draw conclusions based on numerical results. In contrast, qualitative studies are more appropriate when the isolation of the research subject is either impossible or misguiding. Qualitative research produces a holistic depiction, often in a narrative manner, derived from research performed in normal conditions – as opposed to controlled, experimental environments (Creswell, 2013). This type of research might include surveys, interviews and case studies and is more suitable when the related study aspires to address an issue in a broad, non-isolated fashion, formulate new theories (Flick, 2009) or is associated

with immature disciplines which lack comprehensive theory that could be tested in experiments.

The purpose of this dissertation is to understand how complexity can be addressed and managed in the context of Enterprise Architecture, as well as discuss and critique best practice guidelines. Consequently, the selection of a qualitative study was guided both by the fact that the discipline is relatively recent, as well as that the anticipated outcome will possibly consist of a combination of existent and new recommendations, based on the findings. Therefore, this research can be primarily described as exploratory in nature, but incorporating descriptive elements as well; the lack of a consistent body of knowledge allows the research to contribute with new ideas for best practice, while also considering analogies with other engineering disciplines. Nonetheless, EA complexity is an existent, current challenge in real systems; in-depth analysis and description of the relevant factors is necessary to shape up new theories.

2.2 Research Strategy

In order to address the issue of EA complexity, research will start with a literature review phase, aiming to identify frequently recommended guidelines. These guidelines will then serve as the groundwork for the fieldwork stage, which will have the form of a case study of the StrathUni mobile application project of the University of Strathclyde. During participation in this project, the aforementioned guidelines will be constantly revisited and examined in a real scenario of enterprise-level application development. Data gathered or artefacts created at this stage, such as baseline and target models as well as observations, notes and feedback by other participants of the project, will subsequently be analysed in order to test prior hypotheses and discuss how these could be complemented, expanded or reshaped.

2.3 Literature Review

The purpose of the literature review segment is to position the current project in relation to where the EA field currently is. As argued by Bryman & Bell (2011), "using the existing literature on a topic is a means of developing an argument about the significance of your research and where it leads" (p. 91). The literature review helps to delve deeper into a more specialized issue within the whole discipline by investigating a broad set of material, furthering the researcher's knowledge and providing a more solid background for the rest of the research process. Additionally, studying acknowledged, peer-reviewed approaches to the subject aids to determine the scope of the study, decide on what is feasible within the available time and resources, help design the research strategy and give insight on how to present findings, accurately pose and support any claims and conclusions derived at the end (Flick, 2009).

The majority of the literature review process is based on peer-reviewed material published in academic journals and books. However, attempting to maintain relevance and be up-to-date, focus is given on sources of the last decade; consequently, a moderate number of material lacking formal publishing, including organisational reports and research-in-progress papers, should also be considered.

2.3.1 Strategy

The literature review stage was performed in a narrative manner. This choice was made because the purpose of the review was to obtain knowledge of what has already been explored by others, but at the same time allow the researcher to be flexible both on inclusion and exclusion criteria and modifying the limitations for the remainder of the research. Bryman & Bell (2011) suggest that a narrative review is usually more suitable for qualitative studies, as theory is in fact an anticipated outcome; it is thus more sensible to avoid rigid theoretical and conceptual boundaries. Nevertheless, selected features of the systematic approach were adopted, in order to facilitate the process and the reporting

of findings; for example, keeping a search audit trail enabled successful search terms to be reused and helped to identify helpful databases and collections (Jesson et al., 2011).

An initial scoping search was carried out by using keywords taken from the dissertation title in the EThOS online collection of postgraduate and doctoral theses. The same search terms, namely "enterprise architecture complexity", "information systems complexity", "IT complexity", "EA planning/management", were also used in five databases; arXiv.org, OpenDOAR, ScienceDirect, IEEE Xplore and ProQuest. This initial search helped to refine terms, including relevant notions like "heterogeneity", "systems engineering", "dynamic systems" and "emergence", as well as wildcard versions (e.g. "architect* complex*"). The material found was read in order to spot the key points, recurring themes as well as to exclude unrelated results. According notes of important notions were kept in a separate document.

Subsequently, relevant sources were categorized according to the main points found in literature; conceptualization of complexity, factors affecting complexity, complexity related to the business and data domains and current guidelines of complexity management. Following strategies incorporated ideas from the Ellis model of information seeking (Ellis, 1997), like backward and forward chaining i.e. searching material cited in the original source (backward) or searching for sources of information which refer to the original (forward). This kind of bibliography search helped to associate the subject with notions initially unknown (e.g. axiomatic design), complementing the use of keywords with searching in entire reference lists from key sources (Booth et al. 2012).

Eventually, a total of 40 sources was deemed valuable and resulted in the reference list for the literature review chapter; most, especially those directly related to the complexity management issue, were published during the last decade, while a few others, dating as early as the late 90's, were only used in a complementary fashion in regards to well-established subjects (e.g. systems thinking).

2.4 Case Study

The purpose of this component of the research is to test the set of guidelines derived from the literature review. As the topic at hand is very much dependent on the environment conditions and the subject of research (i.e. the organization), the selection of a case study element was rather self-evident. The hands-on experience resulting from involvement in the project will provide the insight required to be able to discuss and reflect on existing recommendations and posit additional ones, in order to formulate a new framework.

2.4.1 Structure

The case study took place in the form of a temporary student placement in the Development & Innovation department of the Information Services directorate of the University of Strathclyde, in Glasgow, during the period of July and early August 2015. Involvement in the StrathUni mobile application development project included participation in the scoping, requirements gathering and analysis and business process redesign (baseline and target modelling) phases of developing a service for handling institution-level events information.

As part of the planning stage, preliminary meetings of the development team were held, in order to establish a vision of the scope of work and determine activities. Participation of the researcher in those sessions involved observation and information gathering; as the business case was already discussed and approved before this dissertation had started, those meetings helped to position the present research in context with the StrathUni project. In addition, observation helped to familiarise the researcher with the language of the team (Becker & Geer, 1957), in terms of both formal terminology and informal conventions used in the department (Bryman & Bell, 2011).

Subsequently, a series of four focus groups with stakeholder areas selected to pilot the Events service were held by the StrathUni project team as part of the baseline architecture analysis stage. These helped to ensure the pilot areas' collaboration as information suppliers as well as provide adequate information about the events-related

"as-is" state of each area (e.g. roles, responsibilities, systems in use, redundant activities) and included questions in a semi-structured fashion; certain questions were predetermined and common for all teams, but the information gathered differed greatly in regards to viewpoint and current practice. Thus, unstructured elements (e.g. open, free-flow conversation) were also used to "give insight into what the interviewee sees as relevant and important" (Bryman & Bell, 2011). The focus groups were organised and coordinated by the development team, so involvement comprised of alternating between actively participating and objectively observing the process. The number of participants was held to a minimum (where possible), in order to ensure that only people with key roles were present, facilitating note-keeping. Difficulties frequently associated with the focus group method, such as possible loss of control over proceedings (Bryman & Bell, 2011) were also avoided due to that. Lastly, action was taken to ensure that all participants' perspectives were adequately phrased and recorded (Krueger, 1998).

Lastly, the business process design phase completed the fieldwork stage of the dissertation. Information derived from the focus groups was coded in categories related to their role in the process (process steps, inputs, outputs, sources) and modelled in "as-is" business process diagrams. These baseline descriptions were then analysed and redesigned to represent the "to-be" processes, also in diagrammatic form. Elements from the Lean/Six Sigma methodology were also applied in the process redesign. In order to complement the process diagrams, a set of diagrams depicting data flows and structures relating to each process were also drawn. All modelling was carried out using Microsoft Visio Premium 2010 and its built-in notation, for simplicity reasons; observation of project participants continued to be important throughout this stage, as well.

2.5 Data Analysis

Data gathered in a qualitative study such as this dissertation can be characterised by high volume and large variety; due to the nature of data, it can be confusing to find a way of analysing and drawing conclusions from disparate notes and documents. In addition, as opposed to quantitative research, qualitative data analysis lacks unequivocal principles on how data should be managed (Bryman & Bell, 2011). However, there are two main

general strategies on how to conduct this kind of analysis; analytic induction and grounded theory. The former includes discussing a hypothesis and then iteratively collect and analyse data, until the inexistence of divergent instances confirms the prior hypothesis; the latter also uses an iterative approach of collection and examination, coding data and then grouping smaller categories into bigger ones in order to derive theory (Glaser & Strauss, 1967). For the purpose of this project, however, a more flexible approach was preferred over the definite selection of a specific strategy. This was primarily due to the immature nature of the discipline, both in terms of literature volume and varying acceptance of ideas posed.

Firstly, a set of relevant categories (codes), derived from the guidelines proposed in literature, was created in the form of a list of bullets. Subsequently, the collected data was evaluated in comparison to this set. The emergence of new categories as the analysis stage goes on is also very important, as theoretical background is sparse in the EA discipline; therefore, the strategy of thematic analysis was selected in favour of comparative analysis to allow new concepts to surface from the data (Creswell, 2013). Following the idea that qualitative data analysis does not benefit much from strict rules, the data analysis strategy in this study was again chosen with flexibility in mind, avoiding creating a rigorous codebook before the fieldwork.

2.6 Presentation of Findings

In qualitative research, presentation of results is most frequently carried out in a narrative manner. The main ideas identified early in the study, as well as remarks, comparisons, confirmation and critique of literature sources can only be presented and discussed appropriately in a narrative text. This way, a logical continuity could be maintained, beginning from the early literature review and arriving at the identification of new research gaps and suggestions for future research on the subject. Results of this study also include the business process and data models created, which are presented in the fourth chapter, again with narrative explanation on their structure and content, as well as discussion on their value as an illustration tool.

3. Literature Review

3.1 Defining EA

In order to properly introduce the subject of how to tackle complexity in the field of Enterprise Architecture, we should primarily aim to develop as thorough an understanding as possible of the discipline of EA itself; in truth, as it will be evident in the following section, the notion itself contains almost by definition the struggle to cope with a complex environment.

Presently, literature about Enterprise Architecture lacks consensus regarding a standard, commonly-accepted definition; the holistic manner in which EA addresses the organization is vague enough to permit only subjective interpretations. Early definitions delineate the term as the interrelation between data, hardware, software, and communications infrastructure as well as the organization needed to support and couple these elements (Richardson et al., 1990). While such interpretations cannot be considered mistaken, they are clearly deficient in failing to consider the key role of business functions in EA efforts. In more recent years, this dearth was identified and addressed to a point where the alignment of IT with the strategic business goals of the organization is now commonly accepted as the major objective of every architectural project.

Consequently, we can define EA as a vehicle to develop "the organizing logic for applications, data and infrastructure technologies, as captured in a set of policies and technical choices, intended to enable the firm's business strategy" (Ross et al., 2003). This clearly implies the division of EA into the four distinct architectural domains, namely Business, Data, Application and Infrastructure (sometimes coined as Technology) Architecture. This might seem more comprehensive, and indeed it is; still, neither the interdependencies between these four subdomains nor the need for efficient overarching governance are evidently indicated. A more recent and meticulous definition is provided by (Tamm et al., 2011), in that "EA is defined as the definition and representation of a high-level view of an enterprise‘s business processes and IT systems, their interrelationships, and the extent to which these processes and systems are shared by

different parts of the enterprise." All things considered, it might be more suitable to argue that EA is about the integration of each and every aspect and function concerned with the organization (Bente et al., 2012b, Graves, 2007a, Bradley et al., 2012). Nebulous as this proposition may seem, it ultimately encompasses the true scope of EA.

Provided that we reach an understanding of what is addressed, the question of "what constitutes an EA" still needs to be answered; (Lankhorst, 2005) describes EA as a complete set of principles, methods and models utilised to design and implement every aspect of the four subsets as well as a structure to provide governance. These principles depict possible constraints and decisions of the enterprise and are thus both strategic and architectural in nature; therefore, they aspire to restrict the solution space and provide a roadmap for transformation. However, EA is not about implementation details, being more of a high-level plan. The classic analogy to city planning, used frequently in literature (Aier et al., 2009), (Schmidt and Buxmann, 2011), could explain this as the difference between a city plan and a detailed, individual building blueprint. In any case, one could also define EA in direct relation to the management of complexity. Based on the fact that it is regularly focused towards very large organizations, often operating in a federated manner and/or various geographic locations, (Bente et al., 2012b) claim that the main issue tackled by EA is to control cost and complexity of IT, and thus define it as the application of architectural principles in order to simplify IT management.

Taking this notion into account, it is easy to establish an association between EA and systems thinking. Undeniably, EA can be viewed as a vast, very complex collection of related components (a system) which has a predetermined enterprise-level goal and functions within a certain environment. The system as a whole can be iteratively analysed into subsystems which form a value chain (i.e. a set of activities necessary to deliver a product or service); in such subsystems, represented by the four aforementioned domains, the output of one is fed as input to the next e.g. the requirements resulting from business analysis are used as guidelines for Data and Application architecture. In addition to the perceptible process approach, the application of systems thinking also encompasses a soft approach to systems i.e. the role of people and their ideas and views of the world interacting with the system processes, which is also crucial in relation to EA (Checkland & Scholes, 1990). This system-related perspective of EA can enable the use of

past research regarding complex systems, forming a solid basis to start defining and understanding the concept of complexity in the EA context.

3.2 Defining EA Complexity

The idea of complexity is relevant to various sciences and disciplines and is, naturally, construed in several different manners accordingly; a precise and consistent definition is thus inapt in helping us develop an appropriate understanding. However, it is crucial to determine an idea of what we are attempting to address i.e. what is a so-called complex problem. (Graves, 2007b) identifies complex issues as "intractable problems" which either do not respond to conventional countermeasures or resurface shortly after they do. Due to their lack of standard behaviour, even under the same external conditions, complex problems are not susceptible to prior analysis and mitigation planning. Consequently, it is also unlikely to prevent their occurrence; rather, best practice is limited to detection and efficient reaction.

The role of systems thinking, as mentioned above, is vital in furthering a consistent understanding of the matter at hand. Acknowledging the existence of subsystems as parts of a whole enables us firstly to partition a big problem into several others which comprise a smaller scope; moreover, ideas explored in literature concerning classic system complexity can be put to use in the equivalent EA field.

From a systems perspective, we should identify both structural and behavioural characteristics of complex systems; in terms of the former, a complex system is firstly constituted by a large number of diverse components, whose performance is highly interrelated, as one of them affects the behaviour of several others. (Widjaja and Gregory, 2012) identify heterogeneity as synonymous to complexity, and proceed to delineate its scale as the variety of business and IT component properties; (Schneberger and McLean, 2003) argue that the degree of diversity of the components of a system affects its total complexity more than their number. Still, their research introduces a disturbing and paradoxical issue, posed by the adoption of distributed environments, coined "the complexity cross" (Figure 1); Breaking down a large system into a decentralized

collection of less complex components does not infinitely benefit total system complexity, as different configurations as well as additional interfaces between distributed components have the exact opposite effect on system heterogeneity.

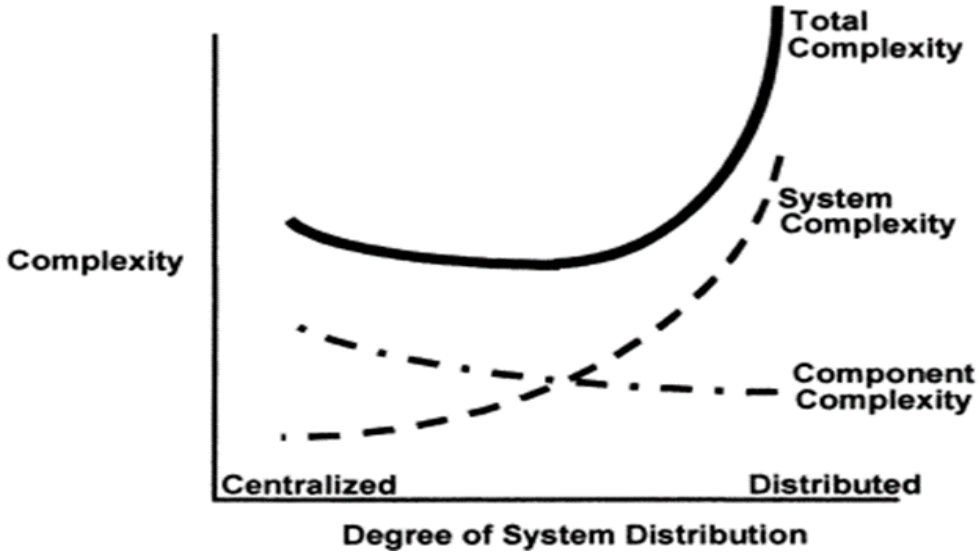


Figure 1. The Complexity Cross (Schneberger and McLean, 2003)

The overall system performance is also influenced by the internal system state at any given time; a complex system will also exhibit both a great number and a large diversity of such states. (Sessions, 2006) argued that "system complexity is a function of the number of states that a system can find itself" and utilised an example of a system of dice; as the number of dice increased, so did the possible combinations i.e. possible system states and thus, system complexity. Finally, a complex system is also characterized by variety of possible inputs and outputs, as well as great interaction with the environment.

In terms of behavioural properties, complex systems are mainly characterised by volatility and unpredictability. Minimal or even non-existent alterations in the environment can lead to different behaviour; moreover, while simple systems display difference analogous to the difference in external conditions, complex systems exhibit discrepancy which is nonlinear to environment changes. Complex systems are also characterised by extemporaneous emergence of structures, high adaptability to the environment and, being highly dependent on its environment, its boundaries are indistinct, making it difficult to be detached and analysed. These features are all applicable to enterprise architectures; the effect of change in large systems' complexity is highlighted by (Schneberger and McLean, 2003) as an even more decisive factor than

volume or heterogeneity. (Saat et al., 2009) argue that the behaviour of complex systems, defined by its initial state and the manner in which the system evolves over time towards another state, can be represented using mathematical modelling of future system states derived by Chaos Theory. Indeed, given its nonlinear relation to change as well as the practical inability to perfectly assess the initial state, the system's behaviour can be described by "deterministic chaos" i.e. evolving deterministically but still appearing to behave randomly.

(Henningsson and Hanseth, 2011) attempted to define the subject using a theoretical model called Assemblage Theory; this theory defines an entity primarily by its "relations of exteriority" i.e. the entity's way of interacting with other entities, rather than the relations between the components forming the entity. Such an assemblage is also characterised by the role its components play in external relations as well as the processes in which these entity elements participate, and is also defined as a dynamic entity, whose components and boundaries are constantly evolving through various levels of homogeneity and clarity, accordingly. This notion, very similar to the idea of "system of systems", is then supported by a case study about the integration of systems used in European trade called eCustoms, breaking down the top-level assemblage (system) into three subsystems i.e. the legal framework, the trade network and the IT layer. The complexity of that latter subsystem is then expressed through the analysis of its dynamic evolution, which involved the integration of systems designed and implemented separately (at national level) or at EU level, or designed with common specifications but implemented individually. The complex structure of eCustoms, the paper argues, originating from lack of governance and high-level design, resulted in the system not only failing to reduce costs and facilitate declaration of imports/exports, but in fact hindering the procedure.

(Schutz et al., 2013b) proceed to define the total complexity of an architecture as a conceptual sum of the complexities associated with each of the four subsets (or subsystems, given that we consider EA as a system), as mentioned above; Business Architecture complexity, associated with the nature of business processes and functions; Information (Data) Architecture complexity is related to how much or how diverse information is needed for business functionality, as well as the amount, variety and level of detail of the relevant data models; Application Architecture relates to complexity

linked with the volume, heterogeneity and volatility of the required software components, as well as their local (department level) or global (organization level) scope; and Technology (Infrastructure) Architecture complexity, associated with the volume and heterogeneity of hardware and software infrastructure solutions (e.g. different vendors, versions, models or legacy systems). These sub-complexities are also complemented by the complications arising from interrelations between the domains; examples of such issues could be found in e.g. how many or which specific applications are needed for a specific business process; how much data is required for applications, as well as their source; how business-driven functional requirements shape data modelling; how application requirements form infrastructure specifications, and so forth.

Yet, depictions of complex systems continue to be as fuzzy as the definition of EA itself; consequently, a common understanding of EA complexity is still a research gap. (Schneider et al., 2014) acknowledged that this variety of definitions impedes progress of the discipline and proposed a comprehensive framework in order to outline complexity in the EA management context. In their work, complexity is identified by four individual dimensions, each comprised by two opposing aspects. The first identified notion is concerned with the amount of variables; a system associated with a large number of unrelated variables of irregular values exhibits complexity that may be subject to statistical analysis. Even if the values of individual variables can't be accurately calculated, the complexity of the system as a whole, referred to as disorganized complexity, may be easier to be addressed using average or mean values. On the contrary, a smaller number of strongly interrelated variables might be associated in ways rather indistinguishable to be analysed, or are improbable to be measured; in this case, the organized complexity present in such a system appears to be more confusing, despite the considerable number of parameters. A second dimension identified in this research is associated with the qualitative or quantitative characteristics of complexity; systems dynamically changing or featuring self-organization and emergence comprise qualitative complexity. Values of examined parameters are influenced by the fluctuations of others or the nature of system components. Quantitative complexity, on the other hand, is linked to the number and heterogeneity of elements and interactions in a system. The third notion of complexity identified by Schneider et al. is related to the viewpoint i.e. is a system inherently, objectively complicated due to a number of factors, thus making

complexity a system property? This might be a dramatically different case than a system which is subjectively identified as complex due to either an insufficient or over-informative system representation or an inexperienced or unfamiliar system observer. The fourth and final dimension delineated is concerned with the structural or dynamic nature of system complexity. In the first case, complexity is associated with system components and their interrelations; in the latter, variations in these interactions as well as the environment of the system cause additional issues of complexity.

The lack of a holistically applicable EA complexity measure have ushered certain researchers to focus on quantification, aiming to conceptualize complexity universally; out of an array of concentration measures, entropy (also applied in software analysis) is the most frequently utilised in literature. (Schutz et al., 2013b) define EA complexity by introducing a heterogeneity measure, comprised of a tuple including the entropy value and the cardinal number of both system components and relations. Complexity is then conceptualized as C= (Number, Heterogeneity), again for components and relations, where Number denotes the instances of components/relations (as opposed to the cardinal number above, denoting the different types of components/relations). Their model was applied in case studies concerned with the use of various types of DBMS and OS in organizations; in the first case, the use of eight different DBMS, disproportionately distributed, would produce the same entropy measurement with six uniformly distributed DBMS; in the second case, the high difference between the entropy measure and the cardinal number of OS in use drove them to the conclusion that certain OS types were used by only a couple of applications; if these migrated to other OS in use, complexity could be reduced. (Schmidt, 2013a, Schmidt, 2013b) also expresses heterogeneity as a statistical property using entropy and argues that the universality of such a measure enables application in different system attributes and levels of detail. (Kandjani and Bernus, 2011, Kandjani et al., 2012a) on the contrary, quantify complexity using a measure called Information Content (IC), which is expressed by the number of relevant environment states that a system needs to be aware of in order to work properly, reinterpreting the IC measure proposed by (Suh, 1999) i.e. the negative logarithm of the probability that the system always satisfies its functional requirements.

However, it is obvious that an expedient understanding and management of EA complexity cannot be based on quantification alone; one must mainly delve into the qualitative factors that cause EA to be complex.

3.3 Why is EA Complex?

In order to effectively proceed to the stage of exploring various approaches aimed to mitigate or reduce complexity, it is valuable to invest some time surveying research relevant to the reasons of existence of this phenomenon; what actually causes all this complexity?

Technology changes are a major complexity factor in information systems architecture; this might entail integration of new technologies/standards, retirement of legacy systems or unsupported software, changing vendors or even regular maintenance and upgrade. These changes are one of the focal points of EA; it is imperative that they are addressed and embraced rather than avoided, as well as successfully explained and communicated by appropriate theory and methodology (Kandjani et al., 2013). (Schmidt, 2013a) also mentions the intensification of legal and regulatory policies as one of the top five EA complexity factors. Lastly, (Schmidt and Buxmann, 2011) as well support the idea that system changes permit growth of complexity when not addressed properly; the authors furthermore posit that the Second Law of Software Evolution, i.e. software complexity increases in time if no specific appropriate measures are taken, can be applied on the more abstract EA level and suggest that, due to this increased scale, amplified complexity might impede the implementation of the changes needed in the EA system.

(Bente et al., 2012c) argue that another reason for IT complexity is the immaturity of software engineering as a discipline. In contrast to classical engineering, software design, development and usage often lacks standard practice or commonly accepted, formal methodology; consequently, impromptu solutions or enhancements are often acceptable. Given that software is subject to regular updates and maintenance, and that it is not tolerable for certain core applications to have large down-time to allow these changes, software that is difficult to maintain can highly affect the EA system complexity. In

addition to the effect of software volatility, (Banker and Slaughter, 2000) also discussed the presence of unstructured software design as a common malpractice adding to the complexity issue. Moreover, the federated nature of multinational enterprises increases their degree of system distribution; the absence of centralized governance inevitably increases redundancy in IT systems, as well as poses issues regarding independently developed, short-term scoped projects at the business unit level (Janssen and Kuk, 2006), (Schmidt, 2013a).

However, one must always keep in mind that the very notion of EA is interwoven with the term "holistic view"; its scope is the structure and behaviour of all business and technical aspects of an organization. Therefore, IS complexity is analogous to business complexity, escalating accordingly in regards to size, scope and purpose. Business architecture is naturally influenced by forces beyond the EA system boundaries, such as socio-political and economic factors, highly competitive and volatile markets as well as legal and environmental regulations; the framework produced by the according strategic analyses perplexes business functionality, especially when related to large enterprises depending on numerous core processes on a global scale and often intermingled in changes like mergers and acquisitions or regular introduction of new business models, products and services. Budget restrictions as well as rigid time constraints might also lead to improvised, proprietary solutions lacking consistent planning or accurate documentation, seriously affecting architecture homogeneity.

The idea of social complexity, posed by (Op't Land et al., 2009), is another key complexity factor – often overlooked in academic research, even though progressively the discipline is shifting away from the IT-centric approach. The aforementioned holistic viewpoint of EA also involves a multitude of stakeholders, coming from a variety of cultural backgrounds, possessing different skills and abilities and holding different interests and functional roles within the enterprise. None of these interests can be deemed more justifiable than others in EA management; it is reasonable to perceive that they will be dynamic and ever-changing, subject to a competitive environment (Schmidt and Buxmann, 2011). The main focus, however, is present: alignment of business and technology. Thus, a need for effective communication, common vocabulary and understandable architectural products is identified, albeit associated with a very diverse set of people involved.

Apart from the role of business architecture, the data architectural subset has also risen to prominence as a primary cause of complexity. (Banker and Slaughter, 2000), defining data complexity as "the number of data elements per unit of functionality", argued that software complexity is predominantly dependent on the amount of data that needs to be processed by an application, an amount correlating with the amount of source code. This premise supports their conclusion that total data complexity is analogous to the effort and expenses associated with software maintenance, as previously discussed. (Delic, 2002) highlighted the escalated volume and deteriorating quality of enterprise data as a major IT complexity source; thenceforth, the term "big data" became a buzzword, as more and more organizations aspire to benefit from data warehousing, analytics and business/market intelligence. However, the impact of data in total EA complexity should not be considered solely as a result of introduction of new technologies; the capabilities created by them are primarily driven by shifting business requirements. The next section will further explore how the data subset, being a major market trend and driver, affects EA complexity both in relation to the way it supports business functionality and its connection with technology.

3.4 Complexity in Relation to the Data Architectural Domain

Nowadays, the term "big data" seems to be a catchword for enterprise executives, academics and article authors alike. Indeed, the ever-increasing rate of data growth as well as the business opportunities and insights provided by data analytics offer considerable rationale for the crucial role of data in EA. Analytical data is not the only cause, nonetheless; the integration efforts in federated environments denote that enterprise data (sometimes also referred to as "master data" in literature), namely consolidated information readily needed by multiple core business functions (e.g. client or supplier data), become a key architectural consideration too.

These two factors are illustrative of the data domain being a leading cause of amplifying EA complexity. Business strategy increases the demand for more and more complex data to support enhanced analysis and more efficacious business processes. The challenges posed by this form of massive data are often concluded as the three V's (volume, velocity,

variety); enormous amounts of data, which are gathered at a speed that hinders processing, and that is comprised by diverse formats and patterns originating from diverse sources (McAfee and Brynjolfsson, 2012), (Madden, 2012), (Kaisler et al., 2012). The quantity of information has a direct relation to EA complexity; while beneficial for business process support, large data volumes increase application complexity (as previously discussed) and also pose new infrastructure requirements e.g. in terms of physical storage or data management systems. The issue of speed in data gathering, aggregation and processing also affects infrastructure complexity; commonly used relational DBMS's again might not be adequate to collect data real-time or query fast enough. Network infrastructure should also be reconsidered in regards to transmission bandwidths. Lastly, data diversity is also a major concern; various formats, structures and metadata – these are all translated in additional complexity. (Kaisler et al., 2012) also mention two further factors associated with data which affect overall EA complexity; data value, defined as the usefulness of data in business related decision-making; and the inherent complexity of data, defined as how interconnected and interdependent data elements are in big data structures. Thus, the nature of information might imply that minor alterations in a few elements can have unpredictable effects for the rest of the EA system. Massive data frequently also means larger numbers of users accessing and modifying it – as a result, more sophisticated concurrency control is required.

Efforts to manage data architecture complexity are still very immature. Survey results (Hayler, 2012) show that less than a quarter of the participating organizations try to consider data quality, and even fewer interpret this issue in terms of costs; it is of course self-evident that low quality data (often described as "dirty") boost complexity. In addition, inconsistent data at enterprise level also poses difficulties; results from the same survey showed that an average large organization have more than five different systems generating enterprise level data, such as customer or product records; as the number of such systems typically scales with the size of the enterprise, some survey participants had almost 100 master data generating systems. The governance of data is an ongoing battle too, one involving balancing organization politics as well; the stewardship and authority over data can cause heightened complexity too. As shown by the aforementioned research, less than a third of enterprises employ data governance schemes, and in 43% of them these programs were reckoned inept. Lastly, even more

recent surveys (Russom, 2014) take also into account the different data types as complexity factors, especially when considering the increase of semi-structured (e.g. JSON or XML) or unstructured types (e.g. multimedia); in order to accommodate and process such data, the introduction of new technologies (e.g. Hadoop, NoSQL or columnar databases) is required. This also affects total EA complexity by increasing heterogeneity – as relational systems are still mostly predominant for operational use.

3.5 Management of Complexity

The augmented costs and reduced efficiency caused by the issue of complexity present in architectures has driven researchers to explore the development of methods and principles of management of architectural complexity – in structural and behavioural system properties. Part of the literature that was surveyed offer a full set of such guidelines in the form of a complete complexity management framework, often featuring equivalences to established, more mature engineering fields; however, the proposed frameworks are deliberately abstract, both in wording and in spirit, in order to be applicable as widely as possible, accommodating the miscellaneous characteristics of enterprises and their respective environment. This section is comprised by an analysis of such methods and proposed approaches.

One of the main points present in literature, albeit very theoretical in nature, is partitioning. The notion of breaking down either the analysis or design of a complex EA system can be linked to various ways in which this can happen; the aforementioned concept of applying systems thinking to consider EA a "system of four systems", represented by the four architectural domains, is only one of them. The fourth principle suggested by (Schutz et al., 2013a) highlights the importance of all four subsets being considered in order to properly address the complexity issue. (Sessions, 2006) argues that system complexity is orders of magnitude smaller when partitioned, identifying the difference as System States × Number of Components (S×D) - partitioned complexity vs. System StatesNumber of Components (SD) – unpartitioned complexity. The paper proceeds to propose a partition-based iterative architectural approach, suggesting to work in steps iteratively to perform the scoping, design, implementation, testing and

deployment of each partition of the whole project (e.g. each architectural domain or business unit subsystem), before moving on to the next one.

A commonly suggested principle to decrease complexity, which can be viewed as a more specific approach to partitioning, is concerned with the consideration of sufficient viewpoints; (Aier et al., 2009) verbalize this as a "criterion of width", required in EA management. This can be translated to the need for representing appropriate information in an equally suitable manner, for all stakeholders. Their proposed framework, the Business Engineering Navigator (BEN) includes a viewpoint catalogue component in order to ensure that the principle is applied. (Graves, 2007b) also justifies this notion in his proposed Tetradian model, in which the various viewpoints are paralleled to the Zachman framework columns and are summarized in four proposed enterprise dimensions: the aspirational (concerned with the business drivers, purpose and enterprise culture), relational (concerned with market relationships), conceptual (linked with beliefs and knowledge assets) and physical (based on enterprise actions and behaviours). (Saat et al., 2009) identify the consideration of various stakeholder concerns as part of their requirement for different levels of EA planning; in such a case, possible divergence can be anticipated before the design/implementation phases.

The use of different viewpoints is frequently associated with a requirement for adequate stakeholder collaboration. (Schmidt and Buxmann, 2011), in their proposed EA management framework, include both viewpoints (as part of required EA documentation) as well as stakeholder participation and communication; all stakeholders should have access to architectural data, be aware of complexity indicators (system redundancy, service reuse, data management etc.) and consequently be involved in relevant decision making. Moreover, it must be ensured that all issues relevant to EA management are properly communicated in order for the implementation to be effective. (Schutz et al., 2013a) additionally support the principle of collaboration by arguing that the quality of data relevant to complexity measures should be made clear to users, in order to aid them make optimal decisions.

Another general issue of complexity management is concerned with the abstraction/level of detail considered. In general, we can accept the premise that "we can understand complex software systems only when they're nearly decomposable and hierarchic"

(Booch, 2008). There might be a greater chance to be effective if it is possible to have models and system descriptions of varying granularity. The case study which formed part of the research of (Schutz et al., 2013a) helped to formulate the principle that architecture should be analysed in various detail levels, thus enabling the consideration of the different viewpoints previously discussed and helping to identify specific complexity sources in the EA. Consistent target modelling of varying granularity is also backed by the work of (Saat et al., 2009). However, when dealing with a system as vast as an EA, it might be rational to avoid more detail than necessary; the criterion of depth, posed by (Aier et al., 2009), expresses the recommendation to maintain the coarsest level of analysis possible. Architectural scope should be centred on the whole, or sets of related elements; there is little value in the EA management configuring the details of e.g. a particular infrastructure component, apart from the case that high-detail elements or relationships affect system-level decisions or can be reused and thus facilitate system-level design. This is furthermore supported by the Architectural Strategies component of the BEN framework. A similar notion is also encapsulated in the fourth principle of (Bente et al., 2012a): "complex systems cannot be managed at object level, but only at a meta level". Practically, this can be translated in a top-down approach, where management is exercised through architectural rules and principles, involved with basic component functionality (Janssen and Kuk, 2006); from then on, business departments are tasked to apply enterprise policies and make their own decisions related to configuration, design interaction, implementation etc. (Schmidt and Buxmann, 2011) also argue the value of setting architectural principles through their EA planning dimension. Still, it is very important to maintain prudence in terms of abstraction; there exists the danger of formulating policies, models or system descriptions that are too holistic and abstract to be applicable in reality (Checkland, 2002).

Most of the relevant literature also tends to be in accord regarding efforts of standardization as a measure against complexity. The introduction of standards may be translated in various forms; firstly, the adoption of an EA meta-model i.e. a common, standardized vocabulary/terminology, which can help in the required communication of artefacts between stakeholders (Schmidt, 2013a), (Schmidt, 2013b), (Schmidt and Buxmann, 2011). (Aier et al., 2009) encompass this argument by proposing possible extensions of the meta-model component of their framework, in order to make it

applicable to different industries and in enterprises of various size and level of maturity. In addition to the meta-model, a common description language for modelling in all subsets is proposed, furthermore encouraging standardization. A second form of standardization is that of system components or relations; (Schneberger and McLean, 2003) identified this as a way to reduce diversity in system components. In this paper, the authors encourage the use of technical standards (e.g. data formats, applications, networking protocols, hardware, databases) as well as user, design, implementation or maintenance procedures (architectural standards). Standardization of infrastructure elements (e.g. standard platform vendors) or interfaces increase interoperability and thus reduce system heterogeneity (Schmidt, 2013b), (Janssen and Kuk, 2006). It is also important to ensure adherence to agreed standards, through the means of effective EA governance, formalizing processes for decision-making and conducting formal review and approval procedures for such decisions (Schmidt and Buxmann, 2011). Lastly, (Widjaja and Buxmann, 2010) addressed the component standardization issue by presenting it as a mathematical problem (Multilayer Standardization Problem). Their proposed solution to the application of this problem to SOA was a software prototype receiving as inputs i) a service-relationship graph, with services as nodes and relations as vertices, ii) artefact information (such as vendor, implementation cost, functions and business-support utility), iii) information cost estimates for pairs of services in the graph and iv) integration cost estimates for sets of services and platforms. Then, the application uses Mixed Integer Programming to propose an optimal SOA; testing results concluded that the proposed solutions were indeed more efficient than instinctive selections.

Several relevant suggestions on complexity management also include versioning of models as well as the use of repositories in order to enable more efficient model management. (Saat et al., 2009) phrase this recommendation in the context of a requirement named “separation of points in time”; EA planning should support multiple versions of target architecture models, in order to accommodate ever-changing requirements and facilitate control and modifications. This notion is frequently related to the consideration of different lifecycles for models and architectural elements; in the aforementioned paper, the authors also pose a requirement of attention to volatility, according to which EA planning must include analysis on how potential changes in target state models can be carried out, as well as how these changes affect other elements, thus

ensuring that the lifecycle of each model is well-thought-out. The Model Management component of the framework suggested by (Aier et al., 2009) also explicitly recommends both versioning capabilities and lifecycle consideration. Furthermore, the authors discuss the idea of architecture repositories, extending it by proposing that, apart from architectural artefacts, relevant architecture strategies are stored as well. The use of repositories to facilitate model management, similar to the importance of the Enterprise Continuum and the Architecture Repository of TOGAF (The Open Group, 2011), as well as their value in reducing complexity is also supported by (Schmidt and Buxmann, 2011), who mention repositories as an important requirement for both EA documentation (baseline descriptions) and EA planning (target descriptions) dimensions.

(Schneberger and McLean, 2003) identified the notion of a system's "rate of change" i.e. how often and in what scale do these changes take place. They proceeded to suggest a reduction in the number of modifications, arguing that less frequent, batch system updates enable us to study system parameters in more depth and thus, improve planning and decisions. However, (Schutz et al., 2013a) revised that principle; after a case study involving interviews with leading architects, considering the rate of change as individual measurement was deemed insufficient. Rather, a timeline analysis of system changes (and resulting discrepancies in relevant complexity measures, e.g. entropy) helps to maintain a clearer view and shape strategy accordingly.

Complexity management cannot be broken down into a commonly applicable recipe, though. Naturally, the number and heterogeneity of system components should be considered appropriately. (Schneberger and McLean, 2003) suggest the use of techniques in order to simplify the components themselves e.g. packaged software, simplified coding techniques or plug-and-play technologies, as well as taking advantage of any chance of combining components into larger functional units. (Aier et al., 2009) also posit the aggregation of existing technical and architectural elements as a factor limiting complexity. Moreover, the number and diversity of relations between components must also be considered. (Widjaja and Gregory, 2012) frame this as a principle, also distinguishing between relations of similar components and relations between components belonging to different subsets e.g. the relation between a data standard and how this is enforced in the related DBMS. This kind of dependencies must be recorded and kept in the same way as done for components. Lastly, the scale of system distribution

also plays a major role; balance must be maintained between distributed functionalities, whose advantages in flexibility are evident in modern, globalized enterprises, and centralizing elements (e.g. enterprise data or applications in SOA) in order to avoid system redundancy and encourage reuse of services and resources ((Schneberger and McLean, 2003), (Janssen and Kuk, 2006), (Schmidt, 2013b)). However, it is imperative to understand that complexity management is an issue of trade-offs.

(Schmidt, 2013b) argues that decisions regarding complexity are specific to architectural layers and domains. This kind of compromises might have to be made in relation to costs, customer satisfaction, dependence on specific vendors or agility. (Widjaja and Gregory, 2012) present two principles related to complexity-based decision making; firstly, they suggest that EA heterogeneity is largely affected by enterprise strategy (see also (Schmidt, 2013a)). The degree of complexity present in an EA is closely intertwined with the goals and objectives of the business; consequently, the manner in which complexity supports these requirements should be included in the related EA documentation. Secondly, complexity decisions are influenced by certain types of component-specific cost/benefit analyses; for example, in terms of vendor diversity, standardization may reduce interoperability issues but increase dependency. Decisions at component level might easily scale up to affect EA complexity on a higher level.

In any case, it is also crucial to be reminded that there exists no such thing as a simple EA – in terms of zero complexity. (Bente et al., 2012a) argue that this is an important principle of actually addressing the issue; expectations should be clear upfront – there is a required minimum complexity needed for the EA to serve its purpose and deliver business functionality (see also (Schmidt, 2013a)). In this frame of mind, Ashby's Law of Requisite Variety, derived from cybernetics but interpretable in this context as "only variety can absorb variety", also becomes relevant. Consequently, instead of desperately aiming to achieve an unmanageable, costly and exponentially complex goal, it is preferable to enhance the system's aptitude to evolve.

The application of elements originating from the field of cybernetics is explored meticulously in the work of (Kandjani and Bernus, 2011), (Kandjani et al., 2012a) and (Kandjani et al., 2013). In the first of these research papers, the authors introduce the application of Axiomatic Design in EA (see also (Suh, 2001)), postulating that the two

main axioms of this methodology can be followed in this context, complemented by a third principle, forming an Extended Axiomatic Design theory. The first axiom, namely the Independence Axiom, states that Functional Requirements of the design must be remain independent. This means that if and when a functional requirement changes, only one corresponding design parameter is affected; changes in requirements do not have a ripple effect on the whole design. This encourages decoupling of the design as much as possible, curtailing dependencies and limiting the architecture's structural complexity. The second axiom, namely the Information Axiom, advocates that, given the first axiom as a prerequisite, the design must have the minimum Information Content (see also section 1.2) possible. This number is therefore analogous to how complicated the system description (i.e. modelling) is; in other words, the more information the design process needs to satisfy system requirements, the less efficient the design will be.

The third axiom, namely the Recursion Axiom and stating that the system designing a system must satisfy the two original axioms; it therefore aims to complement them by ensuring that the design process itself is as simple as possible. Reducing the complexity of planning and design guarantees that not only a system design is only as complex as required, but that any potential modifications needed will also be of sensible complexity. This is of course in accord with that part of literature advocating the importance of facilitating change in EA. Lastly, the notion of self-design is also supported, as (Kandjani and Bernus, 2011) claim that the possibility of a successful architecture rises if the design is handled by EA stakeholders.

(Kandjani et al., 2012a) and (Kandjani et al., 2012b) proceed to confirm the aforementioned methodology by applying it in Global Software Development projects and in the context of Collaborative Networks, respectively. In the prior, the authors come to the conclusion that striving for simplicity in the control model, along with the least complex design possible, improved chances of efficient projects. This led to the proposal of applying principles of cybernetics in EA management. (Zadeh et al., 2012) argue the appropriateness of this idea by relating TOGAF methodology to cybernetics principles and the Viable System Model (Beer, 1984). (Kandjani et al., 2013), positing that the management of complexity is a regular theme in cybernetics, proposed the Co-evolution Path Model, arguing that, in order to maintain balance on how much variety a system needs, that system must evolve harmoniously with its environment i.e. have as much

complexity as required. To achieve that, viable (where system complexity is relatively close to environment complexity), inefficient (system complexity is greater than environment complexity), vulnerable (system complexity is less than required) and non-viable (critical output variables are outside given limits) states should be identified. Subsequently, a path towards viable states must be designed in each case; inefficient states should be transformed by applying axiomatic design to reduce complexity, vulnerable states should be addressed by reconfiguration of structures or acquisition of resources, and feedback must be present for minor or major changes and updates required. The paper uses mechanisms of the GERAM framework (IFIP-IFAC, 1999) as a proposed way of applying the model in practice.

The final interesting point about complexity management raised in literature is the adoption of elements derived from Chaos Theory. (Saat et al., 2009) used Chaos Theory properties to develop analogies regarding EA planning; one good example is the use of the characteristic of complex systems to be sensitive to their initial conditions. The evolution of such systems depends largely on their initial state; therefore, the authors translate this in the EA context and postulate that the better the current state description of an architecture (i.e. baseline models) is, the more efficient the evolution into the target architecture will be. (Bente et al., 2012a) also mention the notion of "the edge of chaos" in order to describe the difference between too much or too little control over a system. Their principle is that architectures are managed and function most effectively when they balance between order and chaos, deeming both sides less productive. (Janssen and Kuk, 2006) also support this, expressing the "edge of chaos" as "the most productive state of the system, where there is maximum variety and creativity".

3.6 Summary

The purpose of this chapter was to identify a collection of best practice guidelines regarding EA complexity management. A number of different sources were explored and various complexity factors were considered, pertinent to the existence of different technologies, standards and legacy systems, various legal and regulatory constraints, multiple and heterogeneous stakeholder requirements, as well as architectural

complexity derived from intricate business functions, data and incongruous software design methods used.

The related guidelines addressing the aforementioned causes of complexity often proposed in literature can be summarised as follows;

- Consider number and variety in system components and relations; maintain an equilibrium between distribution and centralisation of components/services
- Hold sensible expectations; complexity and business functionality are often analogous, therefore the optimum is not an absence of complexity
- Ensure a clear picture of how the accepted level of complexity is necessary in relation to supporting the enterprise strategy
- Maintain a balance between excessive and non-existent control over an EA; consider the trade-off between creativity and chaos
- Partition complexity as much as possible, take a number of viewpoints into account and consider as many architectural domains as possible
- Involve and collaborate with stakeholders, communicating the issue properly and involving them in decision making
- Consider various levels of abstraction, producing models of varying granularity but avoiding excessive detail
- Encourage standardisation of terminology, modelling language, technical components and architectural procedures (formal design, implementation, maintenance and governance)
- Maintain a repository of architectural products; consider different life cycles for different artefacts by enabling versioning
- Keep control of changes in the EA; maintain analyses of evolution through time
- Decouple functional requirements as much as possible
- Produce a design with only as much information as needed to satisfy requirements
- Simplify planning and design processes

These guidelines formed the basis on which the case study of chapter 4 took place, and will be discussed and evaluated in chapter 5, drawing comparisons with what was observed in the fieldwork.

4. Case Study

This section of the dissertation is dedicated to the case study endeavouring to understand in practice the complexity management best practice guidelines previously identified in the literature review. This part of the research took place as part of the University of Strathclyde mobile application enhancement project, including a one-month placement in the Development & Innovation department of the University's Information Services directorate. The chapter is structured as follows; the first section describes the mobile app project in its entirety, including the related business case and project development approach; next, the scope of the present research within the project is defined; the third section discusses the planning and requirements gathering stages of the mobile app project; lastly, the process and data structure analysis stage is described.

4.1 The StrathUni Mobile Application Project

4.1.1 Business Case

The University of Strathclyde mobile app was first launched in 2011 under the name "mPegasus" as an Apple iOS application incorporating a range of basic functionalities like news feeds, campus maps and library services, among others. Since then, increasing expectations and use of mobile devices drove the continuous development of the app, with the latest version released in 2014, currently named "StrathUni". In the same frame of mind, a new phase of enhancement for the app started in 2015; this, tactical in nature, project is furthermore in line with the University's strategic Information Systems Development framework, related to the transformation of the information systems and services of the institution, as a component of the Digital Campus strategy aiming to provide improved student and visitor experience. The app enhancement project, formally defined as "enhanced provision of a high quality comprehensive, accessible app essential to ensure students and visitors are able to make the most of their time at Strathclyde, in keeping with our reputation as a Leading International Technological University", runs

from April 2015 to April 2016; the benefits anticipated by the enhancement project can be summarised as follows;

- New and improved existing services, both aimed to be more personalised and expedient;
- Improved communication between University and students, in terms of reach, relevance and timeliness;
- Improved feedback mechanisms and data gathering for future development;
- Maintain the technological reputation of the institution;
- Nurture University-wide partnership for development, closely aligned with the collaboration values of the organization.

4.1.2 Project Development

The mobile app project is based on an agile development approach; this methodology prescribes development in short periods ('sprints') of 4-8 weeks, followed by frequent releases at the end of each sprint. This ensures that i) the deliverables remain relevant to the formulated requirements and ii) the development process is much more flexible, making it easier to mitigate possible risks and accommodate changes. The totality of an agile project has a clear, longer-term timescale (1 year in this case), comprised of a large set of ambitious objectives; detailed delivery planning is then carried out before the start of each sprint by prioritising requirements according to resources and requirements. Emerging business needs might also lead to some original objectives being revisited or discontinued.

The planning stage is carried out by the Agile Planning Group, consisting of senior-level stakeholders from areas affected by the project and members of the project team; the Planning Group holds a meeting before each sprint and proceeds to prioritise areas of development. Priorities are then documented and kept available for all project stakeholders. Overall governance is exercised by the Project Board, consisting of senior executives of the business units involved; the Board receives the 4-6 top priorities resulting from Planning Group workshops and decide on 2 or 3 for the next sprint.

Reporting of progress is done both to this board and the Planning Group. The design and development phases are executed by the project team, consisting of a Business Project Manager, an IT Project Manager and an assorted number of analysts and developers, depending on size and scope. Through the entire development process, a Product Backlog is maintained; this contains a list of the services under development, the sprint in which they are scheduled and their progress and is available to all stakeholders, kept in the online document repository.

The mobile app enhancement project is carried out as a collaborative partnership between multiple areas of the institution's Professional Services; the project team and Project Board involves members (and senior executives, respectively) of the Information Services (henceforth IS) and Student Experience & Enhancement Services (henceforth SEES), while the Agile Planning Group is formed by stakeholders from IS, SEES as well as other areas which are either affected or could valuably contribute (Media & Corporate Communications, Estates Services, Students' Association, Information Governance and Developers Forum). Figure 2 visualises the organizational structure, highlighting areas related to this project.

4.2 Scope of Fieldwork – The Events project

The case study took place via a placement in the Development & Innovation team of IS and was focused on Sprint 1 (July 15 – August 15) of the mobile app project. The Agile Planning Group for that sprint, which had met quite some time before the beginning of this dissertation, along with according feedback by the Project Board, resulted in three development areas to be carried out in Sprint 1; Rebranding of the app, Push Notifications service and Events service. The present research was thus centred on the development of the Events service, due to its organization-wide, complex nature – more specifically, on the planning, requirements analysis and process analysis/redesign stages.

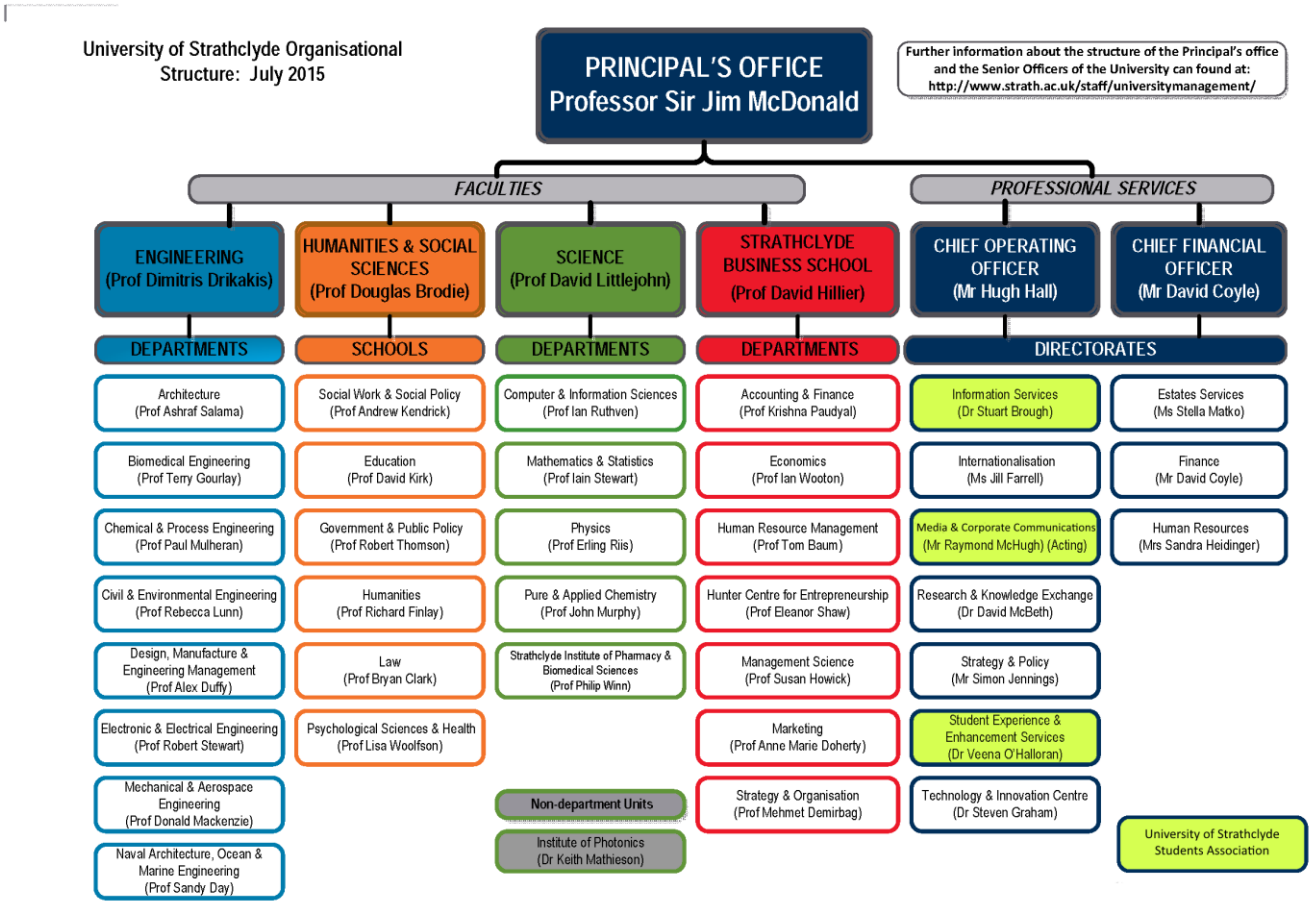


Figure 2. University of Strathclyde organizational structure diagram

The Events project has to do with an institution-level aggregation of information regarding University events, in order to develop an events listing functionality in the mobile app. From an EA point of view, this can be translated into:

- A need to identify how various business units currently manage Events information, leading to analysis and (potentially) redesign of according business processes, along with documentation of baseline and target
- A need to identify and document the various and heterogeneous baseline data definitions, principles, types and sources, across contributing areas, as well as define the "to-be" Events data architecture of the app

These two domains of focus, along with the initial scoping and planning stage, could be parallelised with a partial iteration of the TOGAF ADM (see Figure 3), comprising the Architecture Vision, Business Architecture and Data Architecture stages and occasionally overlapping to the Application subset (in terms of applications used for management of

relevant data). Naturally, all these stages were continuously driven by a central Requirements element. In addition, four viewpoints were considered, corresponding to the four pilot areas selected to contribute Events information: SEES, IS, USSA and Media & Corporate Communications.

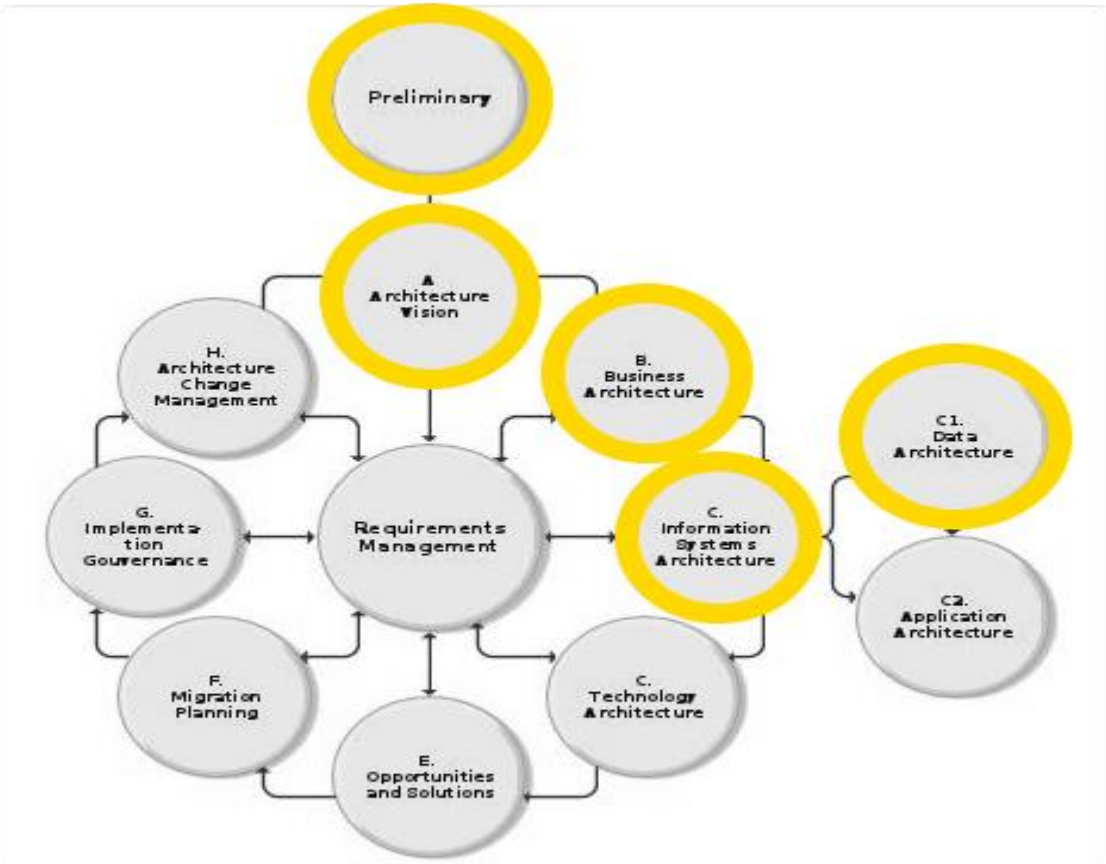


Figure 3. Relationship to the TOGAF ADM cycle

The four selected areas, when further functionally decomposed into teams, comprise a wide array of currently different events-handling processes, systems, data types and definitions. The aim of this project is to provide an integrated events display functionality, centred on the mobile app, while also avoiding drastically transforming everything and thus risking stakeholder buy-in. Therefore, the Events project case study reflects reasonably well the issue of architectural heterogeneity, as delineated in chapter 3 of this dissertation, while also posing a very common trade-off which needs to be addressed effectively.

4.3 Planning and Requirements Gathering

As briefly discussed in the previous section, both the Agile Planning Group and the Project Board had already met in April and May, before the start of this research, in order to plan and prioritise activities for Sprint 1, including (but not limited to) the Events functionality. Consequently, the part of the planning stage in scope for this dissertation began with a project team meeting on June 1, in order to discuss the scope of Events in particular.

4.3.1 Initial Scoping

First of all, results from the Project Board voting procedure were presented. The Board members assess each of the top six priorities selected from the Agile Planning Group, using a scale of 0-10, for each of the following criteria:

- Relationship to strategy (0 for loose relationship, 10 for direct alignment, weight multiplier: x9)
- Impact on customer satisfaction (0 translates to "customer might notice", 10 for customer demand, weight multiplier: x9)
- Time needed to address issue (0 translates to "more than 1 month", 10 translates to "10-15 days", weight multiplier: x7)
- Resources available (0 translates to "none available", 10 translates to "committed resources", weight multiplier: x8)
- Data available (0 translates to "non-existent", 10 translates to "strong variable data", weight multiplier: x7)
- Measurability (0 translates to "complicated/expensive to measure", 10 translates to "easy/low cost measurability", weight multiplier: x7)

The final score is weighted based on the importance of each criterion, as shown in the brackets. Normally, 2 to 3 priorities emerge as focus areas for the next sprint. The presentation of prioritisation decisions based on the Planning Group and Board served as a vehicle of governance, highlighting the importance of Events as a key development

area for the forthcoming Sprint 1. Subsequently, the scope of Events had to be decided by the team internally.

First of all, it was made clear that the target Events functionality would only involve an aggregation of event listings; an events management system (involved with e.g. bookings) was deemed tremendously infeasible for a small agile team, working in a 4 week sprint. Instead, it was decided that the app would include a thin solution, gathering and then displaying basic event information as well as offering a link to the according external web page for more details and functionality. Therefore, the starting point of this procedure was to decide on a simple question: how should we define an event? A standard definition, covering business areas across the institution was not available e.g. the Careers service could classify a seminar as an "event", while Catering services could characterise a weekly meal deal as a special "event" too. Related to the need for an agreement on attributes defining an event, a set of event categories also had to be decided, in order to enable the app to provide filtering functionality. Both of these decisions were agreed to be made in collaboration with the pilot areas selected to be the first suppliers of event information, as parts of the requirements' gathering.

In addition, it had to be made clear that event ownership would be out of scope; information ownership and moderation would remain a responsibility of the supplier, and the app would not store data locally. This was important in order to establish boundaries of organizational roles between business units and the development team; moreover, since an explicit framework regarding this type of information was not present, an issue of data ownership and storage had to be resolved. Lastly, four pilot areas were selected to be contacted as primary suppliers: SEES, IS, Media & Corporate Communications and Students' Association; this decision was based on which areas are most likely to be involved with a higher volume of events.

4.3.2 Planning Workshop with Pilot Areas (User Stories and Use Cases)

The next step was to hold a planning meeting including the Project team and representatives from each pilot area; first of all, participants were briefed on the Events project objectives, scope and timescale. As part of that procedure, three user stories were

created in order to demonstrate realistic use scenarios posing issues that the Events project aimed to address. These can be found in Appendix A.

Subsequently, the process analysis plan was discussed; a series of focus groups with each area was agreed to take place before the start of Sprint 1, in order to document baseline (if available) and target processes and discuss current and "to-be" event categories. Preliminary feedback on how each pilot area currently manages events information was gathered, leading to a brief overview of systems currently in use and identification of responsible stakeholders who should be contacted and involved in the focus groups. Participants were also presented with the capture tool already implemented by the team; this included a simple online form, through the institution-wide Pegasus platform, which enabled suppliers to provide event information to the app. As part of that, a test set of categories was also formed, in order to be cross-checked with the pilot areas during the focus groups. Therefore, a crucial decision for pilot areas and the project team was whether each area would switch to using the new tool or continue to use existing systems. In the latter case, a number of thin services which would extract data from the various systems had to be implemented. This certainly represented a trade-off of heterogeneity management, left to be further discussed throughout the focus groups.

This first analysis also included a consensus that top-level event categorisation would be necessary in order to include a filtering feature; on the other hand, more detailed categories (e.g. target audience for an event) would be desirable, but only included if possible in regards to time and resources. Moreover, the Events definition was finalised, as participating stakeholders identified four basic attributes characterising an Event; Start Time, End Time, Title, Location, Description.

Lastly, the workshop was concluded with a first requirements gathering exercise. A set of use cases formed by the Project team, representing potential functionalities, was presented to the participants; discussing in groups of three, each use case should be characterised as one of the following:

- Must have – part of the Minimum Usable Subset (MUS) of requirements, meaning that this functionality is critical regarding the timely delivery, safety or legality of the project, or that the project will not serve the Business Case if requirement is not implemented

- Should have – the requirement is important, but the project is viable without it, using alternatives or some kind of workaround
- Could have – desirable but less important feature
- Won't have – not important enough to be currently included, the requirement might be moved to next stage of development

This technique, called MuSCoW exercise, along with the formation of concise and communicable use cases, helped to involve non-technical stakeholders in the procedure, translating heterogeneous needs into clear requirements for the project team. The full list of use cases used in the planning workshop, along with the resulting priority scores, can also be found in Appendix A.

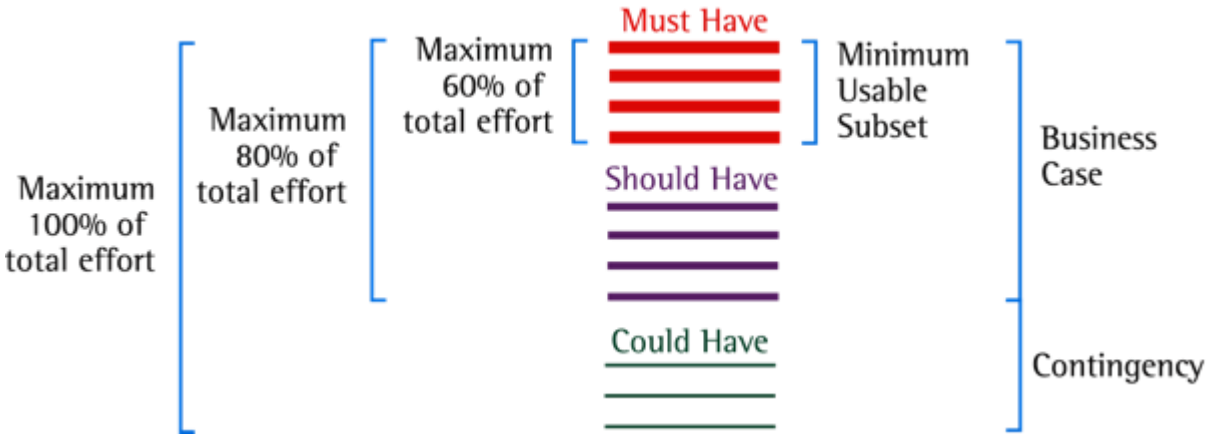


Figure 4. MuSCoW priorities in relation to the Business Case

4.3.3 Parallel Planning – Agile Planning Group for Sprint 2

In parallel to the requirements gathering procedure for Sprint 1, the second development cycle also commenced. In line with the project's agile methodology, the planning process for Sprint 2 had to be streamlined in order to accommodate the fact that numerous stakeholders are typically on leave during the months of July and August. Therefore, in order to maximize participation, the Agile Planning Group for Sprint 2 was organized and held on July 1. While not directly related to the Events project, the Planning Group procedure is briefly discussed below, in order to provide some insight on the methods employed in agile development.

The Planning Group meeting played a twofold role; firstly, it served to inform senior stakeholders of the progress of the entire mobile app project. Subsequently, participants were involved in the planning for Sprint 2. A list of potential development areas, derived by Project team meetings, student surveys and staff feedback, was made available on a noticeboard. Next, a slight variation of the MuSCoW technique, called Planning Poker, was used – in order to make the process more time-efficient and thus encourage future participation; stakeholders were again split in groups of 3 or 4, but instead of having to agree on a consensual characterization of each requirement, each participant received a number of votes: 6 "Must" votes, 2 "Should" votes, 2 "Could" votes and unlimited "Won't" votes, represented by post-its of different colours. Then, while able to discuss with the rest of the group, each proceeded to vote at will by placing post-it's on the noticeboard list. Additional feedback, remarks or ideas for development not present on the list were discussed with the project team and other stakeholders, and were documented on the spot in order to be included on the list in future Planning Groups.

Thereby, the whole Planning Group meeting takes a simplified approach and finally lasts less than one hour – 45 minutes in that case – making senior stakeholders, whose schedule is usually crammed, eager to participate again in the future. The final scores are recorded and weighted accordingly, and the results are kept online in the project repository.

4.4 Business Process Analysis and Data Modelling

Following the formation of a list of requirements regarding Events, the baseline processes and data structures related to Events were discussed and analysed in a series of focus groups with each pilot area. Participants were selected and contacted based on the feedback derived from stakeholders present in the planning workshop; in general, anybody whose responsibilities include any activity related to managing events were contacted. Subsequently, business processes were redesigned accordingly in order to reflect the change in events management brought by the app functionality. This section is structured as follows; first, one of the tools and techniques applied in this stage of development, the SIPOC process map, is briefly discussed as a small introduction; next,

for each of the pilot areas, first the focus group and all related activities are illustrated in detail and subsequently, the “as-is” and “to-be” business processes and data structures are discussed extensively. Only the process diagrams and data model of the first pilot area are presented in this chapter as illustrative examples; the rest can be found in Appendix B.

4.4.1 The SIPOC Process Map

The SIPOC map (or diagram) is a tool applied before the redesign of the business process, used to identify the key aspects of the process improvement phase and thus helping the team develop an understanding of the process before attempting to enhance it. The acronym stands for:

- (S)uppliers – who is responsible for providing inputs to the process?
- (I)nputs – what is the nature of those inputs? Are there exact specifications related to them?
- (P)rocess – how can the process be mapped into steps?
- (O)utputs – what is the result of this process?
- (C)ustomers – who receives these outputs? Do they have specific requirements?

The SIPOC map is usually depicted as a table containing one column for each of the above elements. The process is usually described in a short narrative manner, and is then further presented in a more detailed manner in a business process diagram.

The value of the SIPOC map is particularly prevalent in a services-based organization, where inputs and products of a procedure are not tangible; therefore, this tool assists the development team in comprehending the roles and responsibilities of people involved in the process, as well as providing a top-level process description which is as universal as possible. Thereby, analysts avoid ending up with a number of various perceptions of the same process, resulting from a number of various stakeholders engaged with it.

4.4.2 Media & Corporate Communications

The Media & Corporate Communications area was one of the first to be selected as a pilot for the Events project, as its responsibilities include liaising with external entities or academic staff interested in hosting a University event, as well as the publishing and promotion of such events internally and externally. The pertinent focus group took place on July 6; three stakeholders, representing the Marketing, Conferencing & Events and Publications teams, took part in the activity. No representative of the Internal Communications team was able to be present, however, the rest remained confident that no major disagreement would surface from their part.

Driven by the SIPOC technique, a list of relevant questions were prepared before the meeting; these aimed to answer the questions posed by the SIPOC as comprehensively as possible, in order to provide a broad understanding of their events management process. These questions are presented below:

- Which teams are managing events/handling events information?
- How do these areas currently handle events?
- What is the frequency of events?
- How do they receive events info?
    - In what format? (e.g. structured diary)
- Who are the suppliers of events info?
- What systems are used to supply events info?
- How often are these systems updated?
- Who is responsible?
- What documents are used to provide events info?
    - e.g. calendars, weekly digest
- What rules, constraints, legal/ethics considerations apply to events management?
- What are the challenges currently? What could be improved in current processes?
- What kind of event outputs do you aim for?
- Who are your events for? To who are they directed to?
    - Do you have events targeting specific audiences?
        - Do you distinguish between staff/students?
        - Are there categories of staff/students for targeting audiences?
- Are there (top-level) categories of events?

In due course, however, a more semi-structured approach was selected; instead of adhering to an interview form, the questions served only as rough guidelines to an otherwise free-flow conversation. This decision had to be made on the spot, during the first part of the focus group where the Events project had to be explained to the participants; it was obvious that participants had no clear view of neither the relevant current processes, nor of how the mobile app functionality would be able to help their job. Therefore, involvement in open conversation helped the emergence of ideas of what could be improved, as well as opinions about the "as-is" and potential "to-be" states. Notably, the Conferencing & Events representative specifically mentioned that the "as-is" process was not helping to reach students effectively, and that the introduction of the Events functionality in a mobile app primarily focused on student audiences addressed an actual business need of their department.

Each of the teams actually had a different way of managing events; the Marketing team used the Microsoft SharePoint calendar to create internal events and then invite members of staff accordingly; the Conferencing & Events team received events information primarily via email, and occasionally provided an online form for events requiring registering/booking details; lastly, the Publications team also received events information via email or gathered it from the University website, and proceeded to format it and create invitations or entries in the Weekly Digest for university staff.

Stakeholder diversity was evident throughout the focus group process; participants representing Marketing and Conferencing & Events, being of a younger age, were more open to discussing a completely new process, possibly involving them using the Pegasus capture tool implemented specifically for events in the app. On the other hand, the Publications stakeholder seemed very reluctant in adopting new technologies, and mostly remained hesitant to contribute throughout the meeting. Ultimately, it was decided that the process should change as little as possible in order to still benefit from the introduction of the app: all teams would email event information to the Marketing team, which would now use the SharePoint calendar for all Media & Corporate Communications events; the app would then pull the information directly from SharePoint through an appropriate web service.

The above information was kept in offhand notes during the meeting and later drove the creation of the SIPOC diagram for Media & Corporate Communications Area depicted in Figure 5, while the baseline and target business processes are depicted in Figure 6 and Figure 7, respectively.

Suppliers	Inputs	Process	Outputs	Customers

• Publications Team
• Conferencing & Events Team
• Internal Communications Team
• Staff
  ○ Academic staff
  ○ Internal staff
	• Events info through:
	  ○ Email
	  ○ Telephone
	  ○ SharePoint calendar
	  ○ Weekly digest
	  ○ Facebook/Twitter
		Aggregation and publishing of events	Published events info:
			• Weekly digest
			• Invitation emails
			• Facebook/Twitter
			• MyPlace notifications
			• SharePoint calendar
				• Students
				  ○ Undergraduate
				  ○ Postgraduate
				  ○ International
				• Staff attending
				  ○ Internal
				  ○ External
				• Staff supporting

Figure 5. Media & Corporate Communications SIPOC diagram

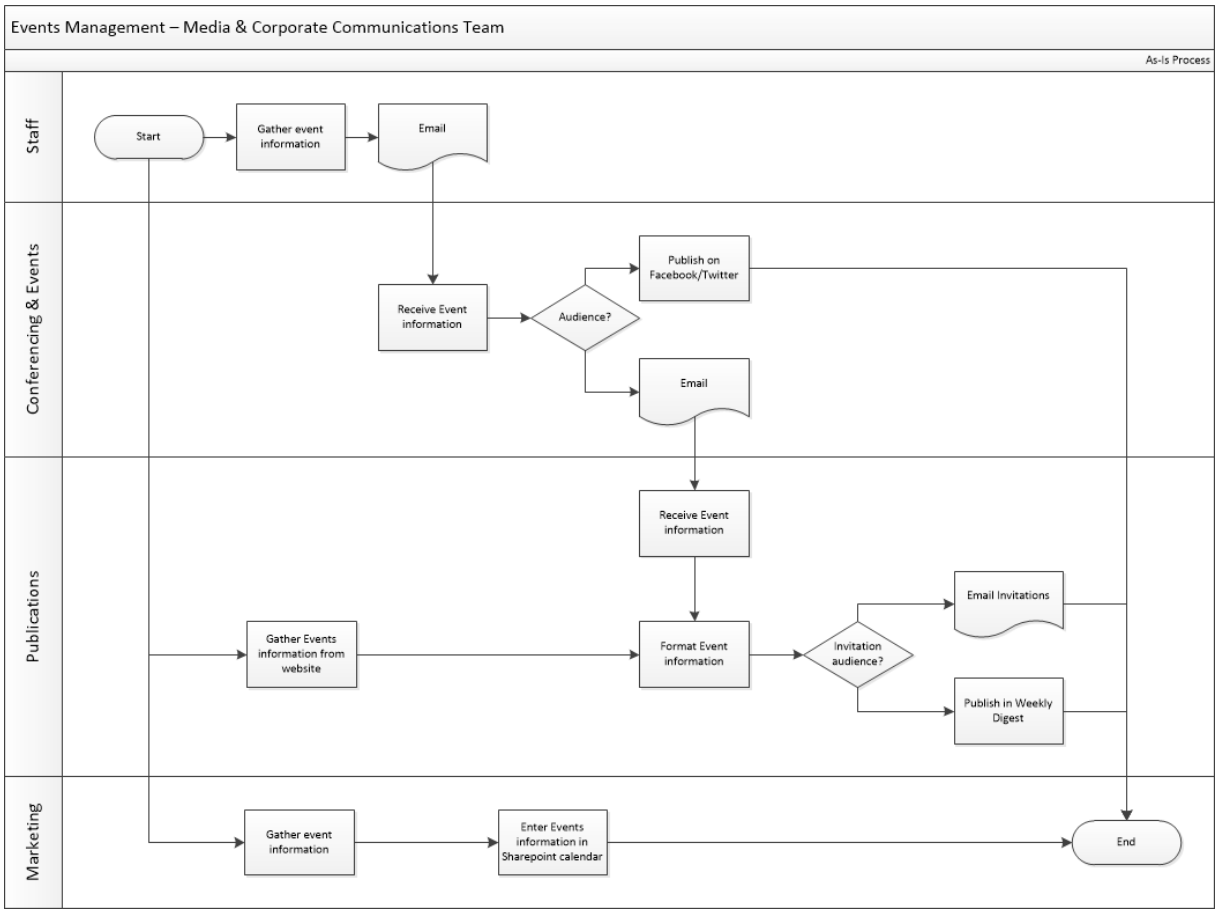


Figure 6. Media & Corporate Communications "As Is" Events Process Diagram

In terms of data structure, the SharePoint data fields currently used were in agreement with the attributes of the basic event definition; as for categorization, the Marketing stakeholder provided a list of categories available as values in the relevant SharePoint category field. However, analysis on former events data by the Project team revealed that these were very rarely used; in most cases, the person creating the event came up with an ad hoc categorization. As a result, existing categories had to be rationalised in order to determine if they were suitable for a student audience; 9 categories (2 from the original SharePoint list and 7 from the ad hoc values) were chosen to be mapped to the equivalent implemented app categories. Finally, it was decided that one more data field would be added to the SharePoint functionality, in order to indicate if the event was intended to be displayed in the app. The suggested data structure is depicted in Figure 8.

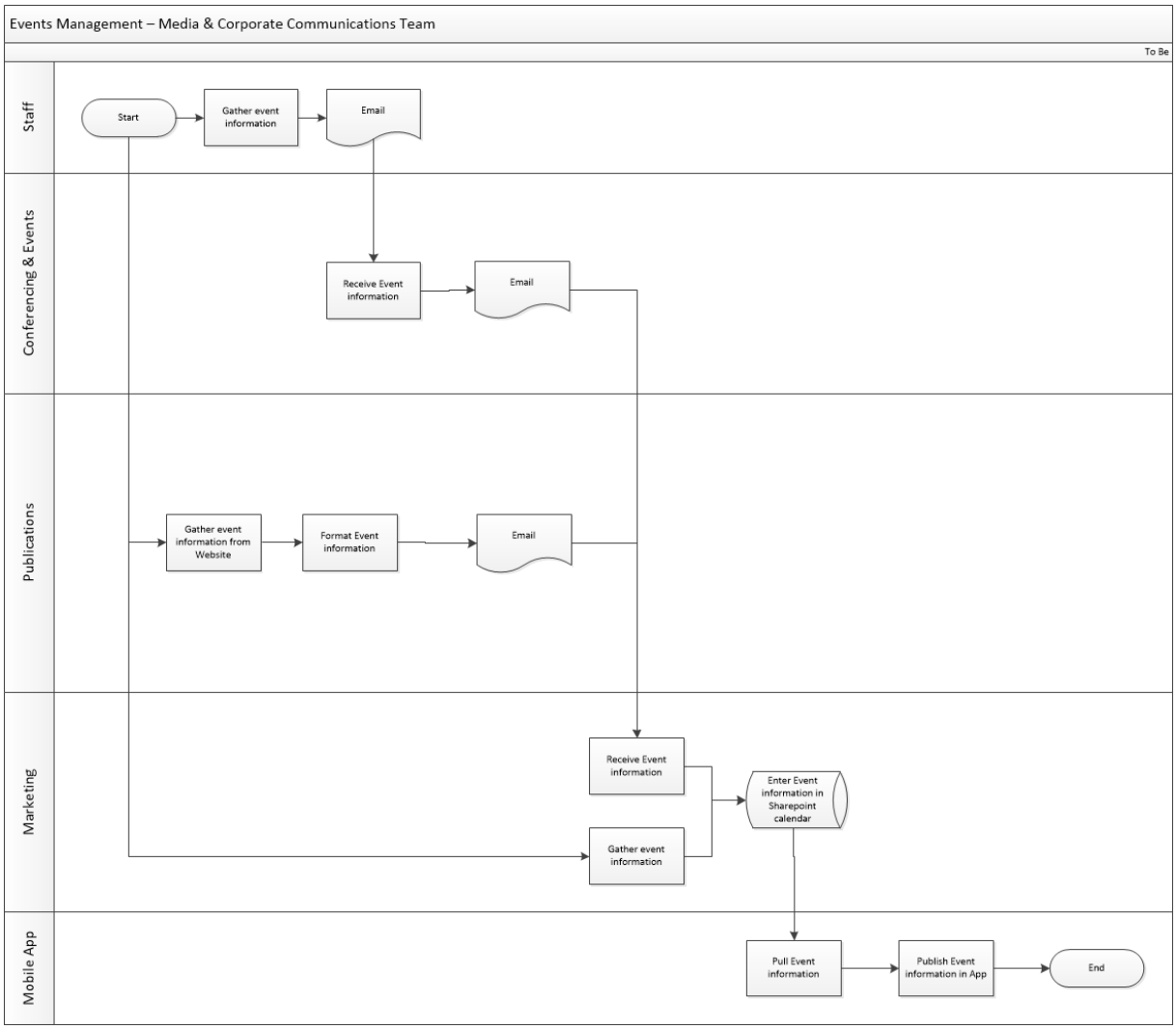


Figure 7. Media & Corporate Communications "To Be" Events Process Diagram

Service	Fields	Suggested Fields	Data Type	Default Values	Example Values
SharePoint Calendar	Title		Text (Single line)		
	Start Time		Date & Time		
	End Time		Date & Time		
	Description		Text (Multiple lines)		
	Location		Text (Single line)		
	Category		Choice	{Meeting, Work hours, Business, Holiday, Get-together, Gifts, Birthday, Anniversary}	{Business, Conference, Dinner, Freshers, Open Day, Reunion, Seminar, Students, UCAS Event}
		Display in App?	Yes/No		

Figure 8. Media & Corporate Communications Data Structure

4.4.3 Student Experience & Enhancement Services

The SEES area was another rather obvious choice for an Events pilot area, primarily due to its direct emphasis on the organization's customers i.e. the students. Multiple teams within the SEES area are regularly responsible for events focused on student audiences, albeit varying in nature and purpose; therefore, the related focus group was the largest and most complicated of the four, and as such, a more structured, formal meeting approach had to be taken. The focus group took place on July 12; 8 business stakeholders attended the meeting, representing the Careers (2 participants), Disability Services (1 participant), Equality & Diversity (2 participants), Widening Access (1 participant) and MyPlace[1] (2 participants) teams. Additionally, one stakeholder representing corporate staff of SEES also participated.

Most stakeholders were unable to attend the preliminary planning workshop, where the SEES area was represented only by the two members of MyPlace, who in turn are also part of the project itself (one developer and one project sponsor/board member). Therefore, a reasonable amount of time at the start of the focus group was dedicated to

[1] MyPlace is the Moodle-based Learning System at the University of Strathclyde. This includes student and staff support functionalities like assignment submission/feedback/grading, class enrolment, provision of class-related materials and notices etc. MyPlace is developed and supported by the SEES area.

explaining the project scope and the agile methodology. This was crucial in terms of sizing up stakeholder expectations – indeed, most of them enquired about the possibility of event booking or other management features. By explaining that the decision for placing such features out of scope was partly made due to the very heterogeneous systems currently used for management/bookings, the Project team also elucidated that the intention of the project is not to replace current practices, but rather aggregate quality data from existing systems, as well as offer a newly-implemented capture tool for any team lacking a process. Thereby, the focus group also served in promoting the Events project internally, making sure that stakeholders were not put off; in the case of the SEES area, this was indeed important, as complexity was higher in comparison to the other pilot areas: more people, currently using at least five different systems, were involved. Therefore, the risk of stakeholders avoiding to be engaged, preferring to stick to what works for them, was higher.

Subsequently, each team took turns explaining their current process (if available) and what systems are already used. This procedure followed a more structured approach, although, again, a set of predetermined questions was not essential. However, open conversation in such a large focus group would complicate things unnecessarily – at least while trying to document process details. Despite the fact that SEES was wholly regarded as a pilot area, in fact each of the comprising teams managed events in a completely separate manner, and the project team had to make sure not to miss out on any details. The SIPOC map for the SEES area is depicted in Figure 9.

In short, baseline processes for all teams are characterised by email inputs or gathering event information locally within each team; subsequently, the Careers team use the University's Enhanced Web Development Service (EWDS), which is offered by IS and provides the design and hosting of stand-alone websites, linked to the main university domain. The web service used by Careers offers an online form to input event information.

Suppliers	Inputs	Process	Outputs	Customers

• Careers Service
  ○ Careers Information Services (MyCareer)
• Equality & Diversity Team
• Disability Services Team
• Widening Access Team
• Directorate level staff
	• Events info through:
	  ○ Email
	  ○ SharePoint calendar
	  ○ eTOBS
	  ○ EWDS
	  ○ MyPlace
		Aggregation and publishing of Student Experience & Enhancement Services events	Published events info:
			  ○ Weekly digest
			  ○ Invitation emails
			  ○ MyPlace notifications
			  ○ SharePoint calendar
			  ○ Website
				• Students
				  ○ Undergraduate
				  ○ Postgraduate
				  ○ International
				• Staff attending
				  ○ Internal
				  ○ External
				• Staff supporting

Figure 9. Student Experience & Enhancement Services SIPOC diagram

Equality & Diversity, Disability Services and corporate SEES staff use eTOBS, the University's online system for booking training courses; while not originally intended for this kind of use, a lower volume of events (compared to e.g. Careers) allowed these teams to utilise it without any trouble. The Widening Access team reported an absence of a standard process, managing events mostly through email and occasional use of SharePoint. Similarly to the prior case of Media & Corporate Communications, the goal was to alter current processes as little as implementation cost permitted; in coordination with the developers, it was therefore decided that two API's would be implemented, each intended to pull data from EWDS and eTOBS, respectively; the Widening Access team was encouraged and, in fact, content to use the Pegasus capture tool.

Regarding data attributes, both EWDS and eTOBS were in accordance with the agreed basic structure of an event, featuring only minor issues to be resolved when pulling data (e.g. both systems feature separate data fields for Start Time/Start Date and End Time/End Date, which must be consolidated into two – Start Date and End Date). However, both systems featured data attributes that represent information not intended to be displayed in the app; for example, EWDS contains information regarding event booking which is beyond the scope of the app project. Hence, these attributes would either be ignored by the API when pulling data or scrapped before being passed to the app.

Conversely, the matter of categorization was slightly more complex; the EWDS form included 6 category fields, in order to classify according to event type, provider, audience, faculty, subject and series (i.e. events that are part of a greater sequence of events). As the app originally intends to provide only basic categorization in order to enable filters, selected values sourced from these 6 fields would be consolidated into 3 major category fields – event type, event host (comprised of values from "event provider" and "faculty"), and target audience. On the other hand, eTOBS featured 3 category attributes (i.e. provider, category and type) which, surprisingly, did not contain enumerated-type values; instead, users would specify ad hoc, string type values.

4.4.4 Students' Association

The University of Strathclyde Students' Association (USSA) was the third major choice for a pilot area, selected again as regards to event volume and reach; in March, for example, a total of nineteen events were hosted on the USSA website, oriented to literally the totality of the students. Similarly to the SEES case, USSA also maintains a direct relationship with customers, therefore efficient event management is a key priority. Additionally, the majority of USSA events is also hosted by students, individually or as clubs/societies; thus, students have an interchanging role of both customers and providers. Lastly, since USSA is not part of the organizational structure of the institution, being an external entity, it does not benefit from the related information services in terms of reach (e.g. USSA events are not promoted through university emails). Consequently, USSA is very likely to be the pilot area which would benefit from the Events functionality of the new app more than any other. The SIPOC diagram for USSA is depicted in Figure 10.

Despite the importance of this particular pilot area, the related focus group was simple, concise and rather informal; only one stakeholder needed to attend, whose role is web development and whose responsibilities include the gathering of event information and creation of related posts on the USSA website.

Suppliers	Inputs	Process	Outputs	Customers

• Student Clubs/Societies/Groups
• External businesses
• Unaffiliated students
• VPDA (Vice President of Diversity & Advocacy)
	• Events info through:
	  ○ Email
		Aggregation and publishing of Students' Association events	Published events info:
			○ USSA Website
			○ Facebook/Twitter
				• Students
				  ○ Undergraduate
				  ○ Postgraduate
				  ○ International
				• Staff supporting

Figure 10. University of Strathclyde Students' Association SIPOC diagram

Having attended the early planning workshop, he was fully aware of the project scope and purpose; therefore, both the project team and the stakeholder quickly decided that, since the USSA website would continue to feature event information, there was really no need for a redundant procedure of re-entering data in the capture tool; instead, implementation of an API to extract data directly from the webpage would be much easier.

The "as-is" process was rather simple, as expected; individual students, clubs and societies as well as other external organizations contact USSA via email. Then, the web developer gathers event information and publishes it both on the website and related social media accounts. The current process diagram is depicted in Figure 16. The target process would only change regarding the app pulling data from the website (Figure 17); apart from that, there was no reason to alter the existing process. As for the data structure, the USSA website form offers distinct fields for basic attributes i.e. title, date, description, location and external URL. As USSA events are frequently hosted by student societies or external businesses, marketing requirements also presented a need for displaying images on the events app page; this was easy to accommodate, provided that the event image is hosted externally, so the app can link to it by the image URI. Categories were not currently used by the USSA area, thus it was decided that the default set of categories of the app would be used initially, and if not adequate, it would be reconsidered in future development cycles.

4.4.5 Information Services

The last pilot area chosen for the first development phase of Events was the Library and Information Resources team of the Information Services area. A variety of events, including library tours, drop-in sessions for new students and IT training courses, both online and in person, addressed to staff and students. In the preliminary planning workshop, the Library team representative reported that the area currently lacked a process regarding events handling; therefore, adoption of the Pegasus service for supplying event information was agreed.

The focus group was initially planned for July 16; however, it had to be rescheduled for mid-August, as the related stakeholders first and the project manager subsequently were unavailable due to summer leaves. The meeting finally took place one week after the end of sprint 1, with two Library members of staff joining the project and IT managers of the team. Detailed process analysis was, in general, not deemed necessary; instead, a short but thorough demonstration of the Pegasus capture tool was made in order to ensure that stakeholders were still in agreement to adopt it. Therefore, this particular meeting had less elements of a focus group or interview, serving most as a follow-up confirmation of what was informally agreed in the planning stage.

4.5 Summary

The mobile app enhancement project is firmly based on the strategic priorities of the University of Strathclyde: that is, to maintain the reputation of the organization being an innovative and technological leader, as well as improve student support by offering rich and accessible information on mobile devices. As part of this endeavour, the Events functionality was planned, designed and implemented in an agile approach, throughout the first development cycle (Sprint 1). Regarding the Events project, four areas of the organization were selected to pilot the service, by assessing the volume and quality of event information that these departments were able to offer. This approach, dictated in

part by the agile methodology, helps to encourage future buy-in by other stakeholders by providing a real functional deliverable as early as possible.

In the previous sections, the planning, requirements gathering and analysis/design phases of the Events project were thoroughly discussed. The project scope included only rudimentary event information being displayed in the app, for reasons related to both simplicity of use and development time/resource constraints; this information includes event title, start/end date and time, location and description (also typically including a URL leading to a more detailed web page of the event). The Events project also aspired to provide a filtering functionality, in order to allow users to select what kind of events they prefer to see; therefore, the app also had to support some categorization of events. A basic set of categories was created by the project team, and was subsequently cross-checked with the pilot areas and extended accordingly. Requirements and functional boundaries of the project are reflected in the related user stories and use cases developed.

Consecutive meetings with the four pilot areas shaped a roadmap for the implementation of the app by providing insight into the business processes and data structures currently used for events. In order to ensure collaboration, the project team aimed to alter existing processes as little as possible; it was therefore decided that it was plausible to implement four lightweight interfaces, one for each system presently in use (EWDS, eTOBS, Microsoft SharePoint calendar, USSA website). These API's will extract data from the current system and feed it in the app. For teams lacking a baseline process, a capture tool, based on the institution-level Pegasus system, was also implemented by the development team. Nevertheless, one certain issue did arise before the design was passed on to developers: the eTOBS system was scheduled to undergo extensive changes by the end of 2015. Therefore, to avoid implementing an API which would very soon be useless, this certain aspect of development was put on hold; instead, users of eTOBS were encouraged to use the Pegasus tool for events that they would like to be featured in the app.

The outcome of the analysis and design stages can also be illustrated in the top-level diagram below (Figure 11). Figure 12 and 13 depict screenshots of the completed Pegasus Capture Event service.

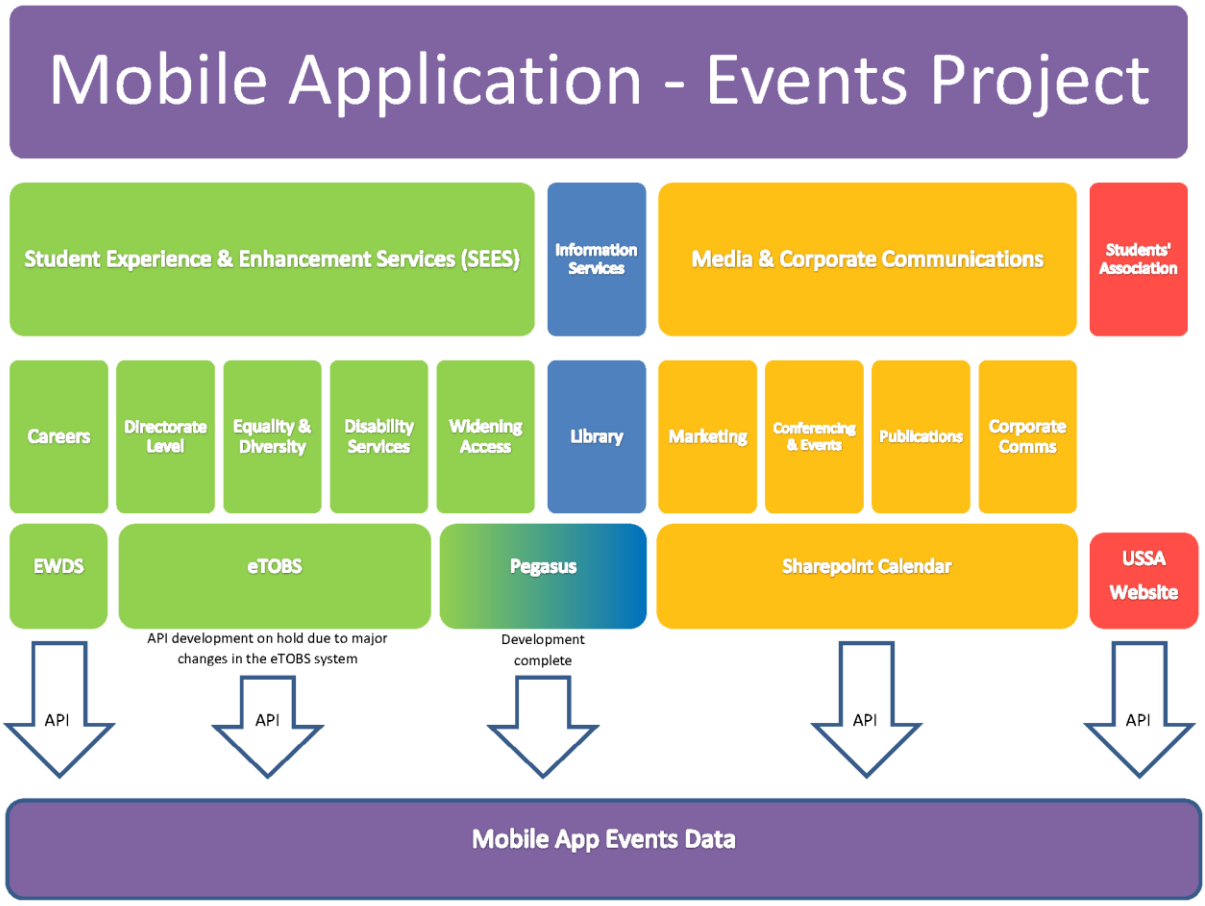


Figure 11. Top-level design of the Events project

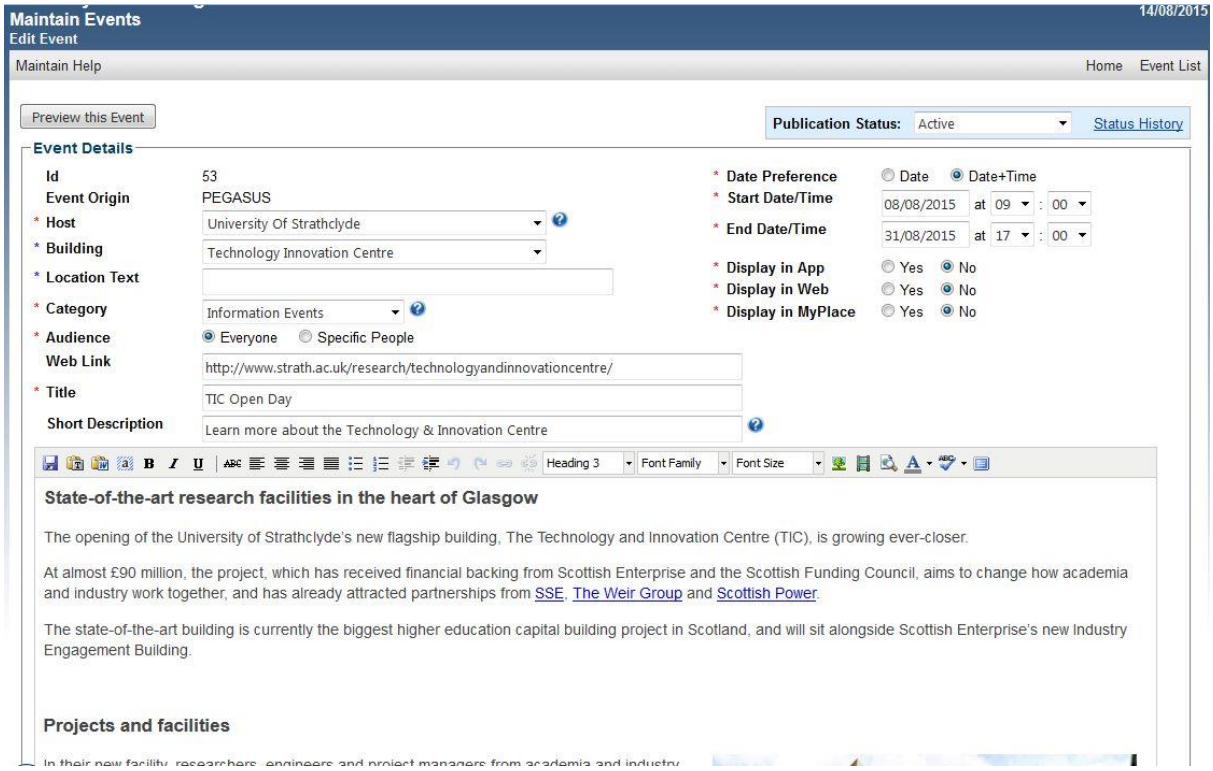


Figure 12. Pegasus Capture Event service – Add Event

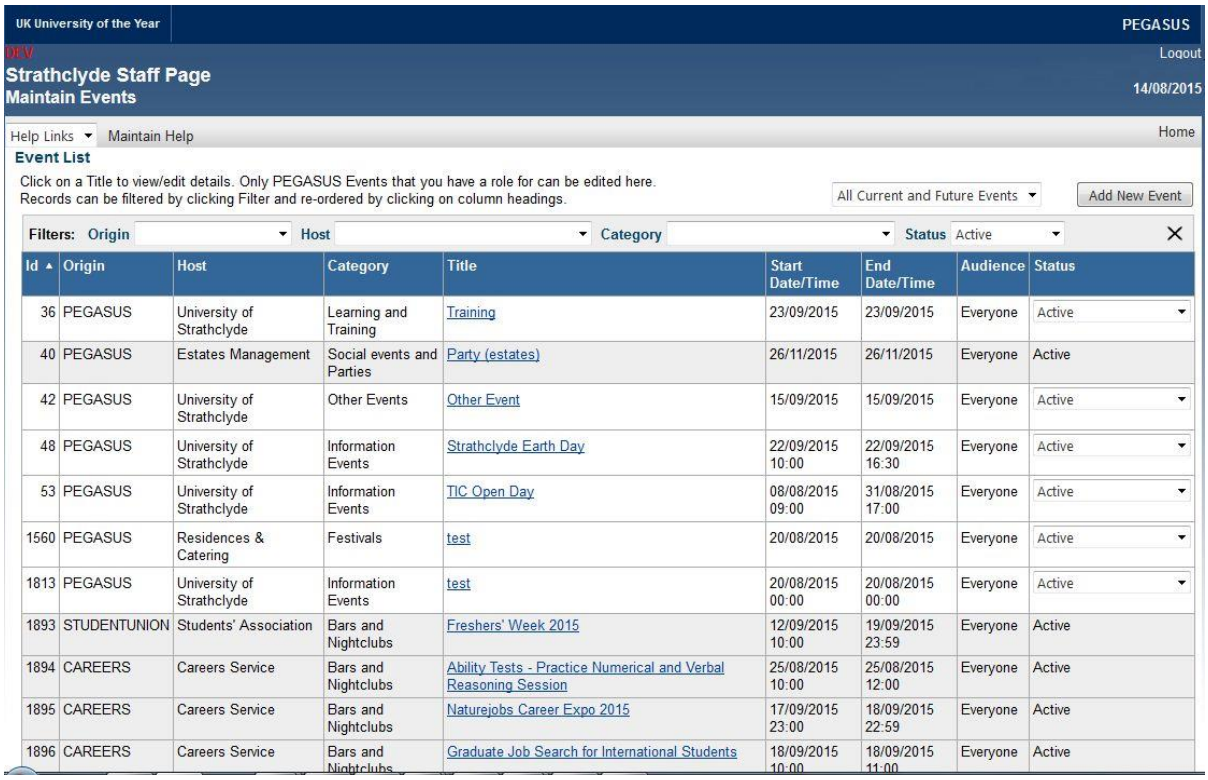


Figure 13. Pegasus Capture Event Service – Event List

5. Discussion

This chapter comprises a thorough discussion of the case study presented in chapter 4 of this dissertation, conducted at the University of Strathclyde Information Services and centred on the mobile application enhancement project. In discussing both the activities and the models created during the planning, requirements analysis and design stages, the present chapter aims to reflect on the techniques used to manage complexity in the case study project by comparing them to the research findings acquired from literature.

The chapter is divided into three sections; the first is focused on the analysis of complexity factors observed during the case study. The second section examines the methods applied to manage complexity by the project team, in the light of the frequently proposed guidelines derived by the research presented in chapter 3. The final section serves as a summary of the discussion.

5.1 Complexity Factors

5.1.1 Technological Heterogeneity

Perhaps the most common complexity factor in EA ventures is technological heterogeneity, resulting from the existence of legacy systems, different vendors or lack of technical standards. In the words of a member of the project team during the focus group with SEES, the University "is a highly devolved organization", and the Events project was indeed direly characteristic of this kind of complexity. Four different systems were currently in use for events management in the pilot areas alone, one of which (SharePoint) also caused issues because of vendor difference: the developer team encountered problems with authentication needed to extract data from SharePoint, due to the use of the Kerberos security protocol. Additionally, the eTOBS system was scheduled to undergo serious overhaul in the last quarter of the year and thus hindered the development of the related API – leaving its users no choice but to shift their process to using the new capture tool. Scaling the events functionality to more departments, as intended, is expected to bring analogous increase in system heterogeneity.

5.1.2 Data Complexity

Closely related to the previous, the issue of data complexity also played a major role in the Events project. The aggregation of large amounts of data not adhering to formal data principles, definitions or a common meta-model significantly affected the complexity of the project. Even though this aggregation involved only basic event data, attribute names and data types were regularly inconsistent: for example, the attribute for "event name" was sometimes termed "title" and allowed for multiple lines of text instead of a single one. As for categories, almost every pilot area that currently used some type of classification had a different set of existing categories, and a few even allowed users to come up with categories on the spot by using a "text" data type instead of enumerated values. The "location" attribute also appeared in some cases as "text" type, while in some systems as

an enumerated class. Data inconsistencies of this sort complicated the implementation stage, as developers had to match attributes representing the same kind of information as well as rationalise current categories to form a set of filters to be used in the app. In addition to all this, the fact that it was the first time that an effort to manage events on an organisation level was made also brought complexities of organisational responsibility between the pilot areas and the IS department. Often, there were no discrete staff roles in regards to who is responsible for event-related actions. Data ownership and stewardship was also a new-found issue since it was the first attempt of aggregating such information.

5.1.3 Social Complexity

The last key element of complexity in the case study project, also commonly found in literature, was the social aspect. Institution-level aggregation of event information can be automatically translated into the involvement of a large array of people, representing various interests and holding different perceptions of the idea of "event" and how it should be managed, but also different opinions on what functionalities this component of the app should include. Project team members are also considered in this, apart from stakeholders coming from the information-supplying pilot areas. The complete set of stakeholders involved was formed by people of different ages, ethnic or educational background and personality types; in a certain focus group, for example, one particular, strongly opinionated stakeholder dominated a good part of the meeting by vocalising suggestions of potential functionalities closely related to her department – or sometimes even relevant exclusively to it. This kind of diversity instigated several communication issues, even within the project team itself e.g. certain team members were happy to proceed with models containing only the absolutely necessary level of detail, while others asked for the models to be as extensive as possible.

5.1.4 Unstandardized Software Development

One of the most commonly identified causes of complexity in architecture is the lack of structured software engineering; designers and developers often resort to impromptu solutions, fixes and patching, steering away from formal procedure or standards. This might be driven from a number of different reasons e.g. limitations imposed either in terms of budget restrictions, time restraints or poorly trained staff members, and is amplified by the fact that software engineering is short of rigid definitions and commonly accepted practice – at least compared with traditional engineering disciplines (Bente et al., 2012b).

The endeavour to enhance the university's mobile app set out to be a tactical project, adopting an agile approach of short sprints of development, lasting a maximum of two months; in sprint 1, when the case study took place, a relatively short, four week sprint was dedicated to the development of three major app features – events, push notifications and rebranding. Therefore, time restraints were present; however, the adoption of a specific SDLC (Software Development Life Cycle) approach (Agile) beforehand, driven by the need to optimise the design and implementation process itself, was largely successful in limiting the effect of this factor.

5.1.5 Unrelated Factors

In contrast, several complexity factors found in literature were not at all evident in the case study. For example, the existence of redundant systems, stemming from the absence of central IT organisation (Janssen & Kuk, 2006, Bente et al., 2012b), is not an issue in that particular enterprise; individual departments do not procure or deploy systems autonomously. Development of IT projects as well as maintenance also take place centrally. In addition, the complexity of the case study project was not associated with business changes related factors often identified in literature, like mergers/acquisitions or regulatory change. However, Bente et al. (2012b) identify competitiveness and the introduction of new services in order to satisfy ever-changing customer demands as a

solid complexity driver; in that sense, the app enhancement project was indeed driven by such causes but still, they did not practically affect development.

5.2 Complexity Management Techniques

The previous section explained the multiple facets of complexity which had to be faced by the app enhancement project – and more specifically by the development of the Events functionality. In order to accomplish this task, a number of tools and techniques were utilised; several others that might also have assisted development are also discussed.

5.2.1 Analysis & Design Techniques

The starting point in complex architecture usually evidenced in similar case studies or other literature sources is partitioning the complex task. As Sessions (2006) postulated, partitioned complexity of a system is undoubtedly more manageable. In the case of the Events project, the first partitioning decision was to initially involve a relatively small number of pilot areas; should this have been different, it would have taken months in order to meet with stakeholders, model baseline and target processes, find implementation solutions for each currently used system and develop the actual code. Simultaneously, as development would be delayed much longer, stakeholder buy-in would be at stake; it is incomparably more efficient to display functional results when trying to persuade people to be involved and adopt the new architecture. Nevertheless, the number of viewpoints (partitions) should be chosen carefully; too many might be widely infeasible, but too little is also problematic. If, for example, only one pilot area was chosen for Events, design and implementation would inevitably have a one-sided approach, facing problems and having to reiterate the whole development process every time a new area was to be involved. Moreover, the data supplied for the launch of the new app version would be sparse and of poor quality.

An observation not explicitly present in the principles derived from the literature, is that attention should be paid in successfully prioritising the selection of viewpoints to be

considered. In the Events project, this decision was driven by a simple question: which departments handle more events than others? This remark can also be applicable in relation to social complexity management; how should analysts decide who should be involved? In the case study, representatives of each pilot area were asked to advise the project team, as previously described. This is most probably preferable to involving, for example, the head of each area by default, as people who are part of the process on a more frequent basis should naturally provide more accurate information.

In addition, the nature of the project itself dictated that all architectural domains should be considered; this, however, was more a result of intuition than a formal decision. Given that this was an app development project, particular focus was given on the Data and Application subsets; redesign of certain business processes happened as a result of the introduction of the new app, while infrastructure was considered only in terms of enabling the implementation of specific functionalities (e.g. bypassing the authentication required by SharePoint). Where possible, separation of duties simplified things greatly; two team members (one analyst and the researcher) were tasked with activities related to business architecture analysis; developers were only involved with the application subset, while also sometimes liaising with the IS infrastructure department, in order to decide the best possible solution for each feature/component; however, certain tasks (e.g. planning and data modelling) had to be done cooperatively. Consideration of all domains is a guideline frequently present in literature (Graves, 2007b , Aier et al., 2009, Schmidt & Buxmann, 2011, Schutz et al., 2013a); indeed, in such projects, it is imperative to deliver a software product adequately supported by the related models but also closely aligned with strategy (Widjaja and Gregory, 2012, Schmidt, 2013a). The StrathUni business case document reflects the relationship between complexity and strategic goals for the entire app reasonably well, however, it is not as clear in subsequent documentation concerning the specific Events functionalities; future development cycles would greatly benefit from such documentation details, especially in sprints taking place later in the app project lifecycle.

The level of detail contemplated when planning and designing is also a major success factor (Aier et al., 2009, Schmidt & Buxmann, 2011, Bente et al., 2012a, Schutz et al., 2013a); ideally, various abstraction levels ought to be considered, resulting in a set of models intended for varying use. Primarily due to the time constraints associated with a

short development sprint, one single level of detail, representing teams within each pilot area, was deemed sufficient for modelling. Overarching, abstract diagrams were only created eventually, in order to summarise the project outcome. Consequently, the final models lacked some detail – some process activities could maybe be further decomposed into sub-processes – which, however, was not deemed vital for the current development cycle. Thus, the goal of avoiding excess detail was accomplished, nonetheless, more fine-grained models might prove very useful for reuse in future efforts of institution-level complexity. Still, the created models were not communicated and cross-checked with the pilot areas; in order for them to serve as a basis for future projects, they should be verified as accurate to real practice.

In the light of the Events project, it is also worthwhile to explore the three EA design axioms, present in the work of Kandjani et al. (2011, 2012, 2013). Although none of them were applied verbatim or as a result of explicit adherence to standard design practice, their inherent value drove the design process intuitively. The first of these rules dictates that functional requirements should be engineered so that each is connected to a single design parameter (Independence Axiom); this rule stood, but only for use cases characterised as "Musts", expressing only core event-related functionalities. For example, use case 14 – "As an event organiser I want to target my event at a specific audience" – represents a key requirement, associated with one parameter i.e. a set of user groups. In contrast, use case 5 – "As a student I want to be reminded about events I'm interested in" – is dependent on the design parameters of use case 6 and/or 23 (adding events to student calendar and registering interest in events, respectively). Extending the axiom to all requirements might prove dysfunctional in practice, especially for features beyond the Minimum Usable Set; it might be more realistic to aim at satisfying the axiom in each development cycle separately.

The second axiom dictates that the design must contain the least Information Content possible. In practice, the design phase satisfied this rule by avoiding excess detail and resulting in models of a "good enough" basis, serving only as a rough roadmap to satisfy basic requirements; indeed, based on feedback provided by the project manager, the researcher revisited the most compound business process models (SEES and Media & Corporate Communications) even after the end of sprint 1, in order to further streamline them before being archived in the repository. Later app development should build on

those by adding detail according to the scope of each sprint. The third and final axiom, i.e. the planning and design process should themselves be as simple as possible, was accurately satisfied. The selection of a specific pilot group played a major role in simplifying both the planning and design stages of the first sprint; the involvement of only two team members (the project manager and the researcher) also facilitated communication through planning and design, as well as collaboration with the developers.

A last overarching observation is that in the Events project case, the process models were not actually used as roadmaps for the implementation stage. The developers utilised mainly the use cases as functional requirements and the data models as guidelines on how to specifically handle issues regarding the development of the API's. The business process diagrams served only as descriptions of the architecture before and after the Events service, and were stored in the repository as reference. This might be either be translated to an existing issue of cohesion between the design and implementation stages, or raise a point about the importance of process modelling in certain software development efforts.

5.2.2 Standardisation

One of the most notable complexity management guidelines proposed is to maximise standardisation in the EA context (Schneberger & McLean 2003, Janssen & Kuk 2006, Aier et al. 2009, Schmidt & Buxmann 2011, Schmidt 2013b). This can be translated into several perspectives, some of which were successfully employed. Adoption of the agile methodology formalised a part of the development process, mainly related to governance, e.g. Project Board and Planning Groups. Furthermore, a detailed product backlog is kept on JIRA page[2]; this contains all implementation issues related with each sprint (new features, bugs, improvements), along with details of each of them regarding staff involved (assignee and reporter), related components (core service, user interface design etc.), completion status and testing information.

[2] http://jira.lte.strath.ac.uk/

However, certain architectural standards were not clearly present in regards to the design process; the tools used for modelling were selected with the criterion of simplicity, rather than adherence to standards. This practice might pose issues in future projects of different scope or purpose requiring more detailed modelling, or if there are changes in staff or organisational structure. Furthermore, the focus group procedure revealed a lack of standards in the enterprise as a whole. Apart from the array of different applications in use discussed before, an absence of a common meta-model was also reported; introduction of a common vocabulary for e.g. data attributes would greatly simplify future projects, even if system heterogeneity continues to be high. Nevertheless, this was not the case with implementation standards, which are clearly documented and stored in the online Confluence page repository[3]. These standards include guidelines for plugin development (e.g. database setup, source control), standard service/service page structure and testing guidelines, among others. Moreover, other technical standards, primarily related to infrastructure elements, are also in place: desktop vendor (Dell), as well as operating systems (Microsoft Windows 7 Professional) are essentially consistent throughout the organization.

5.2.3 Artefact Management

In terms of architectural artefact management, online repositories are used to store technical and support documentation. Continuous management documents, related to project planning, were held on the Information Services SharePoint site[4]; artefacts related to requirements, specifications, implementation details and feedback are held on the app's Confluence page. In regards to Events, these artefacts included user stories, use cases, preliminary mock-ups of several functionalities (e.g. event notifications, filtering), and relevant API documents. The SharePoint site was accessible only by the project team, sponsors and board, while the Confluence page was publicly open for viewing (but not editing).

[3] http://wiki.lte.strath.ac.uk/display/MPEG/mPEGASUS+Home
[4] https://moss.strath.ac.uk/infostratportal/MobileApp/default.aspx

The latter is organised in a hierarchical structure, organising artefacts in categories (e.g. database API documentation, app API documentation, development guidelines etc.) in order to be easy to find. The practice of maintaining online repositories greatly facilitates governance, team communication and artefact reuse, and indeed several earlier resources (e.g. category structure, mock-ups) were frequently accessed throughout the case study as reference points. The repositories are expected to be shortly updated with the process and data models created. A great effort, successful so far, is also made in order to ensure proper, frequent versioning of artefacts, largely due to happen before and after each sprint. Thereby, each subsequent development cycle is expected to be facilitated by the previous ones.

5.2.4 Additional Tools & Techniques

A key method to effectively limit architectural complexity is to ensure the collaboration of the stakeholders involved (Aier et al. 2009, Schmidt & Buxmann 2011, Bente et al. 2012a, 2012c). This holds true both for the project team and the internal customers and project sponsors; in the Events project case, this was truly one of the greatest challenges. Communication within the project team was facilitated by the small size of it, however, the predetermined weekly briefings failed to materialise in practice. Practical difficulties also hindered physical meetings, as several teams under the IS directorate were scheduled to move offices in August; therefore, project team communication took place primarily through email. Despite this, the small number of team members allowed good cooperation all in all. The project board was frequently (on a monthly basis) contacted with a highlight report authored by the project manager and containing the progress of key project milestones, major monthly achievements, change requests and new issues occurred.

In contrast, the situation was quite complicated with stakeholder involvement; even accounting for a moderate number of pilot areas, there were lots of people who had to be contacted, invited to planning meetings or focus groups and interviewed in order to derive requirements and information on baseline architecture. In order to accommodate this, the multiple views were structured on a top-down organisational manner; for each

pilot area, every specific team should provide one view, translated to the corresponding business process (and subsequently represented in the process diagrams). Therefore, teams consisting of several stakeholders present in meetings were encouraged to be in agreement.

The preliminary planning workshop was also a vital success factor, helping to ensure stakeholder participation early on, and also involving the internal customers in decision making; through the MuSCoW exercise, they actually shaped a great part of what the Events service includes. Still, it was essential to ensure collaboration and effective communication throughout the focus groups as well. People were encouraged to come up with opinions on the present planning as well as suggestions for future development cycles. An effort was made to keep the focus groups as concise as possible, usually lasting one hour at most. Lastly, the project team endeavoured to involve only those currently responsible for event management, in order to avoid large and tiresome meetings if possible. Although the success of collaboration will be utterly determined by how much the new Events service will be supported by the pilot areas, requirements and design information gathering was straightforward in comparison to the project's complexity. In hindsight, nonetheless, the project might have benefited from the potential utilisation of a complexity measure, used both within the project team and with stakeholders. A quantification of complexity (e.g. an entropy based measure) could have helped to underline and communicate the necessity of the new service, aiding stakeholders to understand why everything takes place, as well as enabling the project team to demonstrate the benefits of the new architecture, ensuring continuous and future support at an enterprise level. The project team paid specific attention to maintain rational expectations; a certain level of complexity would be present in an enterprise solution, however, a related measure (and possibly a graphic representation of its present and target values) would help to communicate these expectations.

The issue representing the balance between centralisation and distribution in EA is also pivotal in addressing the system's complexity (Schneberger & McLean 2003, Widjaja & Gregory 2012). The Events project was by default an effort of bringing together a set of incongruent event-related information; until now, university events, handled in a completely distributed manner in terms of business process, data structure and application usage, were only disparately available across the organisation. The new

aggregation effort was largely facilitated by the enterprise-wide back-end systems currently available (e.g. Pegasus). The aggregated data feed, currently intended to be used by the mobile app, can subsequently serve as input to other services e.g. the University website or social media. However, it is important to be reminded of two key points: firstly, this kind of aggregation does not centralise neither physical data storage nor data ownership and stewardship; secondly, in most cases the new architecture does not alter the related processes of each department in terms of responsibilities or system use. Therefore, events continue to be managed in a distributed fashion, allowing each area to be quite flexible, while at the same time, there exists a useful single point for any user or system to have access to the data, making it reusable. Centralising event management would not reduce redundancy, as usually intended when reducing distribution, but instead magnify complexity by adding more components and relations to the architecture (e.g. possibly extra staff needed for handling massive data entry, data cleaning or increased server costs).

A complex information system such as the case study example is certain to undergo frequent changes; therefore, the early observation posited by Schneberger and McLean (2003) i.e. that the reduction of change in complex systems limits their overall complexity, is not easy to be applied in practice. In certain cases, changes can indeed be batched, e.g. hardware or operating system updates should be done at the largest scale possible; however, enterprise application development such as the app enhancement project must be closely connected with rapidly altering requirements and external competition. Consequently, related changes, bug fixes and introduction of new features ought to take place more frequently. The upkeep of a timeline change analysis is thus proposed (Schutz et al., 2013a); in relation to the StrathUni project, there does not exist a simple, communicable change analysis. Detailed documentation of previous versions, planning, requirements or implementation-related, albeit helpful, might not be easy to be comprehended by executives. Monthly highlight reports also incorporate a section describing changes made in each particular phase but still, a high-level overview of changes brought by entire enterprise projects could be valuable in monitoring complexity.

5.3 Summary

The Events project addresses the arduous task of creating a single aggregation point for a diverse set of data across the institution. A number of factors caused this endeavour to be characterised by a notable level of complexity; increased technical heterogeneity, as a number of different systems are currently utilised for events management; data complexity, resulting from the lack of principles and meta-models, as well as issues of ownership and moderation between the various business areas involved; social complexity, pertaining the participation of multiple stakeholders holding various interests and priorities, as well as the existence of diverse business domains and processes.

This chapter reconsidered the set of complexity management recommendations in relation to the Events project. The guidelines below were followed, implicitly or explicitly; partitioning the complex task by adopting an array of viewpoints, and also considering all four architectural domains; ensuring proper communication and collaboration with project stakeholders and internal customers; producing models as coarsely as possible, in order to remain on a meta level; maintaining effective and standardised governance procedures; balancing effectively between distribution and centralisation; maintaining artefact repositories; and keeping a realistic objective of rational complexity, providing a level of flexibility and supporting strategy and business functionality. Nevertheless, similar endeavours could benefit by the adoption of standards in modelling and data definitions as well as formalisation of user and maintenance procedures. The maintenance of a top-level timeline of architectural changes could also be of value in terms of communication and planning. Lastly, it is important to note that the recommendation for independence of functional requirements from design parameters might not be realistic in practice, especially regarding optional features which aim to make the product more desirable, rather than delivering what is expected. This is also the case with the suggestion to reduce system changes; in tactical projects such as Events, frequent modifications are not only unavoidable, but probably compulsory for the success of the enterprise project.

6. Conclusion and Future Research

This chapter revisits the purpose of this dissertation, highlighting findings and proposing future research areas concerning EA complexity. It is structured as follows; the first section includes concluding remarks and reflection on the study, starting from the literature review and arriving at the insight obtained by the fieldwork experience; the final section provides a number of recommendations for future study on the matter at hand.

6.1 Conclusion

The purpose of this dissertation project has been to explore further the issue of complexity in the EA context. The literature review conducted at the beginning of the project initially served to properly define the complexity problem in EA by summarising a multitude of related definitions addressing to both quantitative and qualitative features. Subsequently, the first major research question posited and investigated pertains to the factors causing EA complexity. Primary causes include technology changes which unavoidably lead to heterogeneity, a notion almost synonymous to complexity; integration of new technology standards, infrastructure overhauls, decommission of legacy systems or even scheduled maintenance and upgrades are run-of-the-mill procedures in modern enterprises, which however are often overlooked. This is frequently also the case with the second major complexity factor – social complexity. On one hand, there exists a multiplicity of stakeholders involved, holding various interests and roles. On the other, there are numerous business departments and according functionalities, ever-changing in order to enable sustainability, which need to be efficiently supported by technology.

The other main research question concerned the appropriate complexity management techniques; the research produced an array of valuable recommendations which complement the most widely used EA frameworks. These guidelines were centred on increasing technical and architectural standardisation, partitioning architectural efforts

in terms of viewpoint and domain, consideration of multiple (but sensible) level of modelling detail, effective upkeep and attention to life-cycles of artefacts, facilitation of architectural changes, decoupling of functional requirements and simplification of planning and design processes.

These principles were derived from various sources and were thus postulated based on very different scenarios and environments; therefore, the third research question addressed by this dissertation was to understand how complexity is managed in practice. The University of Strathclyde mobile app enhancement project was used as a case study, helping the researcher to comprehend the actual challenges and the manner in which the theoretical guidelines can or cannot be applied – and if so, to what extent. It was found that many of these principles are valid in practice; however, they must be carefully refined according to the scenario. For example, technical infrastructure standardisation is widely unrelated to a project related to the development of an enterprise-level software such as the case study; in contrast, architectural standardisation, related to governance, planning or design, was found to be impressively valuable. This poses the single greatest challenge in addressing complexity: the peculiarities of each EA effort put a rigorous set of principles to being only partially relevant. Complexity management can seldom be driven exclusively by textbook theory; spontaneous trade-offs and intuitive decisions based on experience will always form some part of the struggle.

6.2 Recommended Future Research

Future research endeavours on complexity management can be focused on the development of frameworks for more specific application. Generic EA development methodologies like TOGAF and Zachman can be complemented by various sets of guidelines according to the following:

i) Nature of organisation: EA complexity is too coupled with the business itself. A higher education institution will naturally behave differently than a multinational, federated enterprise trading in a range of sectors. Therefore,

complexity management frameworks could be oriented to specific business areas.

ii) Maturity level: Enterprises manage complexity differently according to which principles they have already applied. An organisation with a highly standardised technical infrastructure level should set different priorities and complexity goals than one whose departments procure on a distributed basis. Consequently, specific frameworks could be developed for enterprises which only recently turned towards EA.

iii) Purpose of project: As previously mentioned, the discipline could benefit by the development of complexity management frameworks specific to each type of EA effort. Examples include enterprise software development, master data management projects, mergers and acquisitions etc.

Furthermore, theoretical research could be focused on proposing and verifying a commonly accepted complexity measure, or possibly a standard method for enterprises to assess and present ICT-related complexity (e.g. system heterogeneity).

6.3 Reflection

The initial goal of this dissertation project was to produce itself a comprehensive framework for EA complexity management. However, it soon became evident that such an aspiration was too ambitious for a three-month MSc project; in order to produce a framework of actual business value, more careful, analytical insight on the business had to be developed – something that would require much longer, continuous involvement in a number of different enterprise-level projects.

Consequently, the dissertation project has yielded a summary of a broader set of best practice guidelines, and then proceeded to evaluate them through the case study. This experience has proven invaluable to the researcher, who was able for the first time to develop a solid understanding of the difference between theory and practice in the Enterprise Architecture discipline. Additionally, the entirety of the research process

greatly cultivated the researcher's critical and analytical thinking, and the related placement served as priceless work experience in planning, design and analysis.

References

Aier, S., Kurpjuweit, S., Saat, J. & Winter, R. (2009) 'Enterprise Architecture Design as an Engineering Discipline '. *AIS Transactions on Enterprise Systems,* 1 (1), pp. 36-43.

Banker, R.D. & Slaughter, S.A. (2000) 'The Moderating Effects of Structure on Volatility and Complexity in Software Enhancement'. *Information Systems Research,* 11 (3), pp. 219-240.

Becker, H. S. & Geer, B. (1957) "Participant Observation and Interviewing: A Comparison", *Human Organization, 16.* pp. 28-32.

Beer, S. (1984) 'The Viable System Model: Its Provenance, Development, Methodology and Pathology'. *The Journal of the Operational Research Society,* 35 (1), pp. 7-25.

Bente, S., Bombosch, U. & Langade, S. (2012a) '*Foundations of Collaborative EA'*. *Collaborative Enterprise Architecture: Enriching EA with Lean, Agile, and Enterprise 2.0 Practices.* Elsevier, pp. 137-157.

Bente, S., Bombosch, U. & Langade, S. (2012b) '*What Is Enterprise Architecture?'*. *Collaborative Enterprise Architecture: Enriching EA with Lean, Agile, and Enterprise 2.0 Practices.* Elsevier, pp. 31-38.

Bente, S., Bombosch, U. & Langade, S. (2012c) '*Why Collaborative Enterprise Architecture?*'. *Collaborative Enterprise Architecture: Enriching EA with Lean, Agile, and Enterprise 2.0 Practices.* Elsevier, pp. 1-28.

Booch, G. (2008) 'Measuring Architectural Complexity '. *IEEE Software* (July/August 2008), pp.14-15.

Booth, A., Papaioannou, D. & Sutton, A. (2012). Systematic Approaches to a Successful Literature Review. London: SAGE Publications Ltd.

Bradley, R.V., Pratt, R.M.E., Byrd, T.A., Outlay, C.N. & Wynn, J.D.E. (2012) 'Enterprise Architecture, IT Effectiveness and the Mediating Role of IT Alignment in US Hospitals '. *Information Systems Journal,* 22 (2), pp.97-127.

Bryman, A. & Bell, E. (2011) *Business Research Methods.* 3rd ed. New York: Oxford University Press Inc.

Checkland, P. & Scholes, J. (1990). *Soft systems methodology in action*. Chichester, West Sussex, England: Wiley.

Checkland, P. (2002) *Systems Thinking, Systems Practice.* Chichester: John Wiley & Sons Ltd.

Creswell, J.W. (2013) *Research design: qualitative, quantitative, and mixed method approaches*. 4th ed. Los Angeles, CA.: SAGE Publications Ltd.

Delic, K.A. (2002) 'Enterprise IT complexity'. *Ubiquity,* 2002 (January).

Ellis, D. (1997) "Modelling the information seeking patterns of engineers and research scientists in an industrial environment", *Journal of Documentation,* 53(4), pp. 384 - 403

Flick, U. (2009) *An Introduction to Qualitative Research.* 4th ed. London: SAGE Publications Ltd.

Glaser, B. G. & Strauss, A. L. (1967) The Discovery of Grounded Theory: Strategies for Qualitative Research. Chicago: Aldine.

Graves, T. (2007a) '*An overview'*. *Real Enterprise-Architecture: Beyond IT to the whole enterprise* Colchester, England: Tetradian Consulting, pp. 1-10.

Graves, T. (2007b) '*KA: Architecture as a way of thinking'*. *Real Enterprise-Architecture: Beyond IT to the whole enterprise* Colchester, England: Tetradian Consulting, pp. 72-79.

Hayler, A. (2012) 'Time to tame data architecture complexity, but task is tough'. *Computer Weekly*.

Henningsson, S. & Hanseth, O. (2011) '*The Essential Dynamics of Information Infrastructures* '. *Thirty Second International Conference on Information Systems*. Shanghai.

IFIP-IFAC (1999) '*GERAM: Generalised Enterprise Reference Architecture and Methodology, Version 1.6.3'*.

Janssen, M. & Kuk, G. (2006) 'A Complex Adaptive System Perspective of Enterprise Architecture in Electronic Government'. *39th International Conference on System Sciences*, 2006 Hawaii, USA. IEEE.

Jesson, J.K., Matheson, L. & Lacey, F.M. (2011) *Doing Your Literature Review: Traditional and Systematic Techniques*. London: SAGE Publications Ltd.

Kaisler, S., Armour, F., Espinosa, J.A. & Money, W. (2012) '*Big Data: Issues and Challenges Moving Forward*'. *46th International Conference on System Sciences*. Hawaii: IEEE.

Kandjani, H. & Bernus, P. (2011) '*Engineering Self-Designing Enterprises as Complex Systems Using Extended Axiomatic Design Theory* '. *18th IFAC World Congress*. Milan, Italy.

Kandjani, H., Bernus, P. & Nielsen, S. (2013) '*Enterprise Architecture Cybernetics and the Edge of Chaos: Sustaining Enterprises as Complex Systems in Complex Business Environments* '. *46th International Conference on System Sciences*. Hawaii: IEEE.

Kandjani, H., Bernus, P. & Wen, L. (2012a) '*Enterprise Architecture Cybernetics for Complex Global Software Development: Reducing the Complexity of Global Software Development Using Extended Axiomatic Design Theory*'. *IEEE Seventh International Conference on Global Software Engineering (ICGSE)*. Porto Alegre: IEEE.

Kandjani, H., Wen, L. & Bernus, P. (2012b) '*Enterprise Architecture Cybernetics for Collaborative Networks: Reducing the Structural Complexity and Transaction Cost via Virtual Brokerage* '. *14th IFAC Symposium on Information Control Problems in Manufacturing* Bucharest, Romania: IFAC.

Krueger, R. A. (1998) *Moderating Focus Groups.* Thousand Oaks, CA: SAGE Publications Ltd.

Lankhorst, M. (2005) '*Introduction to Enterprise Architecture'*. *Enterprise Architecture at Work.* Berlin: Springer-Verlag, pp. 1-10.

Madden, S. (2012) 'From Databases to Big Data '. *IEEE Internet Computing,* (May/June 2012), pp. 4-6.

McAfee, A. & Brynjolfsson, E. (2012) 'Big Data: The Management Revolution '. *Harvard Business Review*, October 2012 pp. 61-68.

Op't Land, M., Proper, E., Waage, M., Cloo, J. & Steghuis, J. (2009) *Enterprise Architecture: Creating Value by Informed Governance* Netherlands: Springer-Verlag Berlin Heidelberg.

Richardson, G.L., Jackson, B.M. & Dickson, G.W. (1990) 'A Principles-Based Enterprise Architecture: Lessons from Texaco and Star Enterprise'. *MIS Quarterly,* 14 (4), pp.385-403.

Ross, J.W. (2003) *Creating a strategic IT architecture competency : learning in stages.* Cambridge, Mass.: Center for Information Research Sloan School of Management, Massachusetts Institute of Technology.

Russom, P. (2014) *Evolving Data Warehouse Architectures In the Age of Big Data.* TDWI Best Practices Report Second Quarter 2014. TDWI Research.

Saat, J., Aier, S. & Gleichauf, B. (2009) '*Assessing the Complexity of Dynamics in Enterprise Architecture Planning – Lessons from Chaos Theory* '. *Fifteenth Americas Conference on Information Systems (AMCIS)*. San Francisco, California: Association for Information Systems. Available at: http://aisel.aisnet.org/amcis2009/808.

Schmidt, C. & Buxmann, P. (2011) 'Outcomes and success factors of enterprise IT architecture management: empirical insight from the international financial services industry '. *European Journal of Information Systems,* (20), pp.168-185.

Schmidt, C. (2013a) '*How to Measure Enterprise Architecture Complexity: A Generic Approach, Practical Applications, and Lessons Learned* '. *The Open Group Conference*. London: Scape Enterprise Architecture Office and Consulting.

Schmidt, C. (2013b) '*Measuring and Managing Enterprise Architecture Complexity*'. *Sebis Workshop on the Complexity of Application Landscapes – Models, Measures, Management (CALM3)* Scape Enterprise Architecture Office and Consulting.

Schneberger, S. & McLean, E. (2003) 'The Complexity Cross - Implications for Practice'. *Communications of the ACM,* 46 (9), pp.216-225.

Schneider, A.W., Zec, M. & Matthes, F. (2014) '*Adopting Notions of Complexity for Enterprise Architecture Management* '. *Twentieth Americas Conference on Information Systems*. Savannah.

Schutz, A., Widjaja, T. & Gregory, R.W. (2013a) '*Escape From Winchester Mansion - Toward A Set Of Design Principles To Master Complexity In IT Architectures*'. *Thirty Fourth International Conference on Information Systems*. Milan.

Schutz, A., Widjaja, T. & Kaiser, J. (2013b) '*Complexity In Enterprise Architectures - Conceptualization And Introduction Of A Measure From A System Theoretic Perspective* '. *ECIS 2013 Completed Research*. Available at: http://aisel.aisnet.org/ecis2013_cr/202?utm_source=aisel.aisnet.org%2Fecis2013_cr%2F202&utm_medium=PDF&utm_campaign=PDFCoverPages.

Sessions, R. (2006) 'A better path to enterprise architectures'. *Microsoft Technical Articles*. Available at: http://msdn2.microsoft.com/enus/library/aa479371.aspx

Suh, N. (1999) 'A Theory of Complexity, Periodicity and the Design Axioms '. *Research in Engineering Design* 11, pp.116-133.

Suh, N. (2001) *Axiomatic design: advances and applications.* New York: Oxford University Press.

Tamm, T., Seddon, P.B., Shanks, G. & Reynolds, P. (2011) 'How Does Enterprise Architecture Add Value to Organisations? '. *Communications on AIS,* 28 (1), pp.141-168.

Widjaja, T. & Buxmann, P. (2010) '*Service-Oriented Architectures: Modeling the Selection of Services and Platforms '. Software-as-a-Service.* Gabler, pp. 219-238.

Widjaja, T. & Gregory, R.W. (2012) '*Design Principles For Heterogeneity Decisions In Enterprise Architecture Management*'. *Thirty Third International Conference on Information Systems*. Orlando.

Zadeh, M.E., Millar, G. & Lewis, E. (2012) '*Mapping the Enterprise Architecture Principles in TOGAF to the Cybernetic Concepts – An Exploratory Study* '. *45th International Conference on System Sciences*. Hawaii: IEEE.

Appendix A

User Stories

International Postgraduate student

Loukas is an International Postgraduate student from Greece and has been studying here at Strathclyde for nearly a year now. He has completed the first part of his MSc in Enterprise Architecture and is now writing up his Masters dissertation. He would like to stay in Scotland upon completion of his degree and during this "self-study" period this is an ideal time for him to start thinking about employment opportunities and applying for jobs as his diary is more flexible now the taught part of his course is complete. He has therefore started to actively look for opportunities with companies across Scotland. After talking to a local student who studies in another department, he is upset to hear he had attended an open recruitment event, on campus, last week with a large local company and in fact had in his possession a department newsletter encouraging ALL students to attend future events of this nature. Loukas had not received any notifications about this event and remains confused as to who to speak to make sure he finds out about the future events the newsletter refers to.

Head of Department

Jean is a Head of Department in the Faculty of Life and after attending a conference at the University of Melbourne she has made arrangements for some students to visit Strathclyde as part of a knowledge exchange exercise. She would like to hold a welcome event for these visitors on campus, but has a limited budget for doing so, as such she would like to advertise the event to students from the Faculty Of Life only to gauge interest and then once she has an idea of numbers potentially open up the event to a wider audience across campus. The visitors are arriving next Friday so she needs to get the information to the correct audience quickly but is unsure how to go about this. She puts a poster on the Department notice board but is really embarrassed in front of her visitors when only 4 students turn up.

Visiting Professor

Professor Cullen is about to begin a visiting Professorship at Strathclyde and has been invited to his new department's monthly evening lecture by email. He is looking forward to the opportunity to meet his new colleagues over some cheese and wine and has been given instructions in the email to visit the department web page for event information i.e. time, location etc. Upon arrival at the campus at 5:40pm and in plenty of time for the 6pm lecture, he struggles to find the building and then the appropriate room (he calls the department contact but they have left for the day). He finds the correct room at 5:55pm only to find a hand written notice stuck to the door stating that the lecture has moved to a building on the other side of the campus. When he finally reaches the new location at 6:15pm he stumbles in red faced to find the only available seat at the front of the lecture theatre and is extremely disappointed and embarrassed to have missed the key note introduction by the Dean - who he had in fact intended to chat with for 10 min before the event began over a glass of finest merlot!

Use Cases

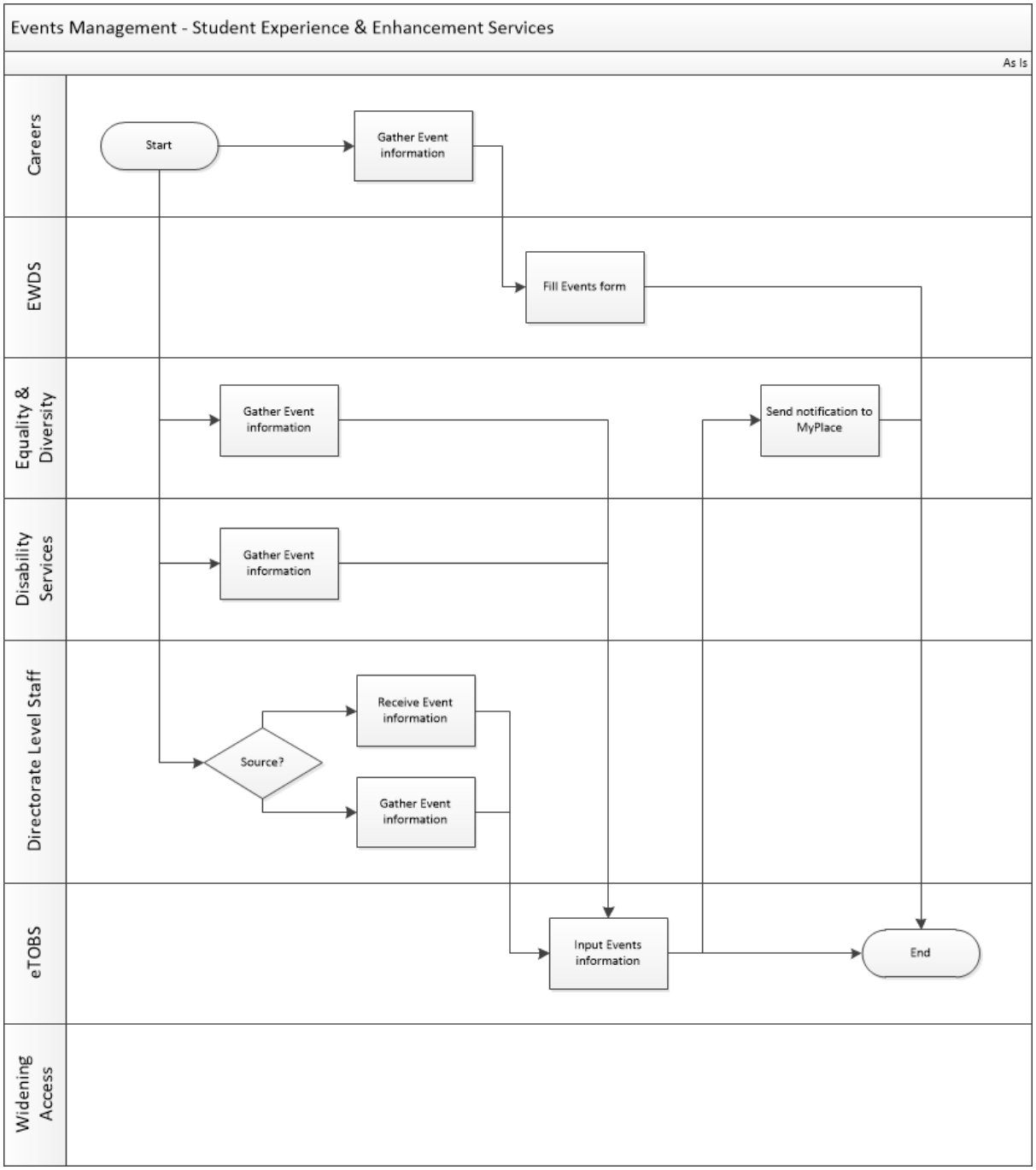
ID	Use Case	MuSCoW
1	As head of department I want to publicize events for my department	M
2	As a [member of staff|student|visitor] I want to see what events are happening in a list	M
3	As a [member of staff|student|visitor]I want to see what events are happening near me	C
4	As a [member of staff|student|visitor]I want to see events within a specific timeframe	M

ID	Use Case	MuSCoW
5	As a student I want to be reminded about events I'm interested in(see 23 & 6)	M (if 23)
23	As a student I want to register my interest in an event (see 6 too)	S
6	As a student I want to be able to add events to my calendar	C
7	As an events organiser I want to advertise a guest lecture	M
8	As an events organiser I want to cancel an event	M
22	As an event organiser I want to change my event's details	M
9	As an events organiser I want to alert users in enough time if details for my event have changed - e.g. time or location (see 5)	M
10	As a student I want to see events from the careers service (and other providers!)	M
11	As a student I want to see information about graduations / my graduation	M
12	As a student I would like to pay for some services through the mobile app e.g. my library fees and topping up my print credits	
13	As a senior manager I want to be able to withdraw an event immediately	M

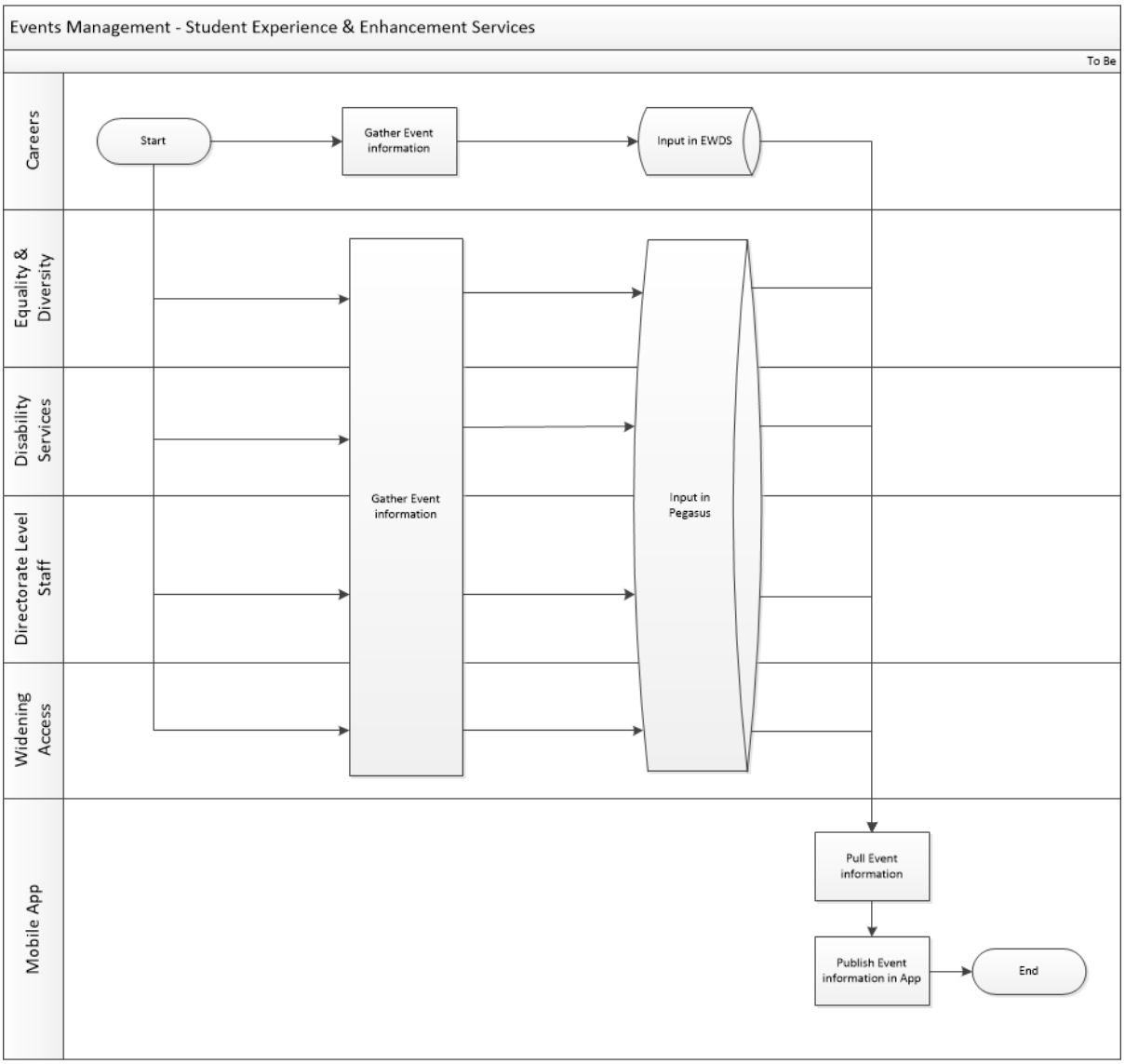
ID	Use Case	MuSCoW
14	As an event organiser I want to target my event at a specific audience	M
15	As an event organiser I want to be able to see event analytics to compile a report	C
16	As a potential event organiser I want to know how to get my event added to the listings	M
17	As an event organiser I want to display information in a visually appealing way	M
18	As an event organiser I want to preview my event as it would appear on the mobile app before publishing	M
19	As an event organiser I want to filter and sort events in the management/admin system	M
20	As a senior manager I want to be able to moderate information	W
21	As a visitor I want to see the location of the event on the campus map	M
24	As a user of the app I want to limit reminders to specific times of day	M (if 23)
25	As a manager I want to be able to grant other users access to manage event listings	M

ID	Use Case	MuSCoW
26	As a "display point" (app, web site, page, digital signage...?) I want to filter what events I show	M

Appendix B



Student Experience & Enhancement Services “As Is” Events Process Diagram



Student Experience & Enhancement Services "To Be" Events Process Diagram

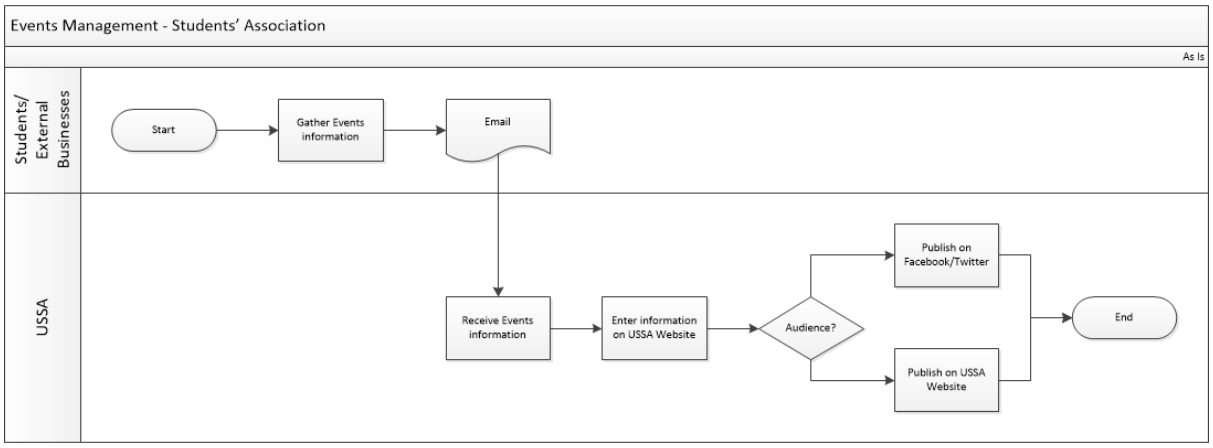
Service	Fields	Data Type	Default Values
EWDS	Event Name	Text	
	Event Start Date	Date	
	Event End Date	Date	
	Event Start Time	Time	
	Event End Time	Time	
	Location	Choice	Values in four lists: University of Strathclyde locations, Other Glasgow locations, Edinburgh locations, Other locations
	Days before event to display on calendar and list	Number	
	Summary	Text	
	Full Description	Text	
	Is Active	Yes/No	
	Allow Registration	Yes/No	
	Places Available	Number	

	Booking Confirmation Message	Text	
	Category: Provider	Choice (Single, Mandatory)	{External, Careers Service}
	Category: Type	Choice (Single, Mandatory)	{Career Fair, CV Tutorial, Employer Presentation, Employer Workshop, Information Session, Insight Day, Networking Event, Online Event, Open Day, Scottish Graduate Fair, Postgraduate Study Seminar, Job Search/Applications/Interviews Seminar}
	Category: Audience	Choice (Single)	{All year groups, Disabled Students, Male Students, Female Students, First Year Students, Second Year Students, Third Year Students, Final Year Students, Graduates Only, International Students}
	Category: Faculty	Choice (Single)	{Cross Faculty, Engineering, HaSS, Science, Strathclyde Business School}
	Category: Series	Choice (Multiple)	{Careers In…, Careers Week, International Week}
	Category: Subject	Choice (Multiple)	{Computing Science, Entrepreneurship, Global, Law, Pharmacy, Work Experience}

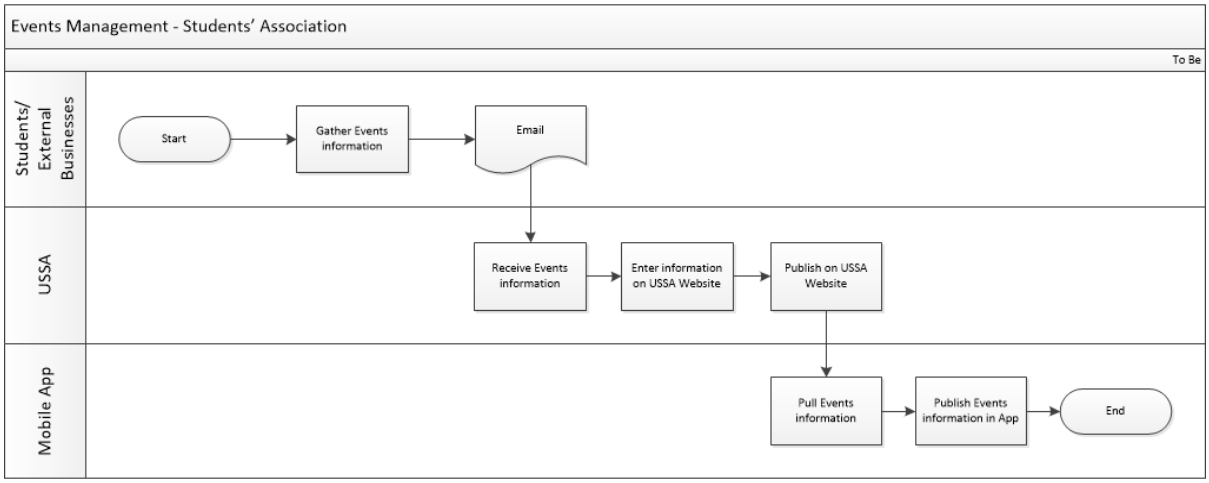
Student Experience & Enhancement Services Data Structure – EWDS

Service	Fields	Data Type	Use
eTOBS	course_title (Table: course_tbl)	Text	unique identifier
	session_code (Table: session_tbl)	Text	unique identifier
	startdate (Table: session_tbl)	“2015-07-22T14:30:27.2496759+01:00”	
	enddate (Table: session_tbl)	“2015-07-22T14:30:27.2496759+01:00”	
	starttime (Table: session_tbl)	"00:00:00.1234567"	
	endtime (Table: session_tbl)	"00:00:00.1234567"	
	location (Table: course_tbl)	Text	
	bookable_status (Table: session_tbl)	Text	
	url (Table: course_tbl)	URL	
	provider (Table: provider_tbl)	Text	unique identifier
	phone (Table: provider_tbl)	Text	
	email (Table: provider_tbl)	Email	
	ds_groups (Table: session_tbl)	Text	
	category (Table: category_tbl)	Text	unique identifier
	courseDescription (Table: course_tbl)	Text	
	type (Table: course_tbl)	Text	

Student Experience & Enhancement Services Data Structure – eTOBS



Students' Association "As Is" Events Process Diagram



Students' Association "To Be" Events Process Diagram

Service	Fields	Suggested Fields	Data Type	Example Values
USSA website	Title		Text	"Freshers' Week 2015"
	Event Date		Date & Time	"Saturday, September 12, 2015 - 10:00 - Saturday, September 19, 2015 - 23:59"
	Location		Text	"Students' Union"
	Summary		Text	"Running from 12th - 19th September this is the only OFFICIAL Freshers' Week line up for Strathclyde.\nFollow #strathfreshers15 or join the FB event here!"
	URL Path		URL	"Freshers"
	Image source URI		URI	"http://www.strathstudents.com/node/37994"
		Event Type		"http://www.strathstudents.com/sites/default/files/CalFW15.jpg"

Students' Association Data Structure