The Synergies between Goal Sketching and Enterprise Architecture

Kenneth Boness
School of Systems Engineering,
University of Reading
Reading, UK
k.d.boness@reading.ac.uk

Rachel Harrison
Dept. of CCT
Oxford Brookes University
Oxford, UK
Rachel.harrison@brookes.ac.uk

Abstract—**This paper introduces a pragmatic and practical method for requirements modeling. The method is built using the concepts of our goal sketching technique together with techniques from an enterprise architecture modeling language. Our claim is that our method will help project managers who want to establish early control of their projects and will also give managers confidence in the scope of their project. In particular we propose the inclusion of *assumptions* as first class entities in the ArchiMate enterprise architecture modeling language and an extension of the ArchiMate Motivation Model *principle* to allow radical as well as normative analyses. We demonstrate the usefulness of this method using a simple university library system as an example.**

Index Terms—**Requirements modeling, enterprise architecture.**

I. Introduction

During project management managers often wish to establish the scope of a project with some degree of confidence as part of stakeholder alignment. Requirements analysis and modeling can be used to help establish such scope, but formal analysis and modeling are often beyond the resources and budgetary constraints of managers working in the real word. As an alternative to such formal analysis we developed an approach called Goal Sketching [1] and used this as a basis for requirements confidence appraisal and scope management [2], [3]. One of the inspirations behind goal sketching is the KAOS goal-directed language and method [4]. However, unlike KAOS the operationalization of goals (i.e. propositions) in goal sketching is effected by products of a project which (in appropriate circumstances) specialize to elements of a system under construction or modification.

Many CIOs are concerned with enterprise architecture, and such architecture has requirements engineering at its core. Our hypothesis is that the ArchiMate[1] modeling language can be very useful to requirements engineers, particularly when used together with Goal Sketching. This paper sets out to show the possible advantages of such an approach.

In order to find the synergies between Goal Sketching and the ArchiMate language we use semantic entailment when modeling aggregation and composition. We also introduce assumptions into the ArchiMate language and a radical interpretation as well as a normative interpretation for its *principle* element.

The contribution of this paper is to provide a simple method of establishing goal oriented requirements together with a supporting defensible argument (via Goal sketching) whilst acknowledging the motivation of stakeholders (via ArchiMate).

This paper is structured as follows. We begin by introducing our goal sketching method. Section III discusses Enterprise Architecture Motivation Modeling, and this is followed by our synthesis of the two methods including our definition of normal and radical analysis. We present an example to illustrate these ideas, followed by related work and a discussion before finishing with our conclusions.

II. Goal sketching

Goal sketching has four types of goal oriented proposition as shown in Table 1. *Motivation* refers to the end desired by stakeholders and *behaviors, constraints,* and *assumptions* are the means by which the products realise the motivation. *Behavior* is equivalent to capability and condition together [5], and is the default proposition; untyped nodes are assumed to be behavioral nodes. *Constraint* is equivalent to constraints as usually understood [5]. An *assumption* is a proposition about the context of the situation-to-be that can be depended upon. Assumptions include domain properties and expectations [4], [6], [7], [8], [9] as well as solution clarifications, simplifications and transformations. The guiding maxims of Goal Sketching are *goal refinement through semantic entailment* and *keep all objectives satisfied*. The root of the goal sketch must be a single superordinate motivational goal.

Structural completeness guarantees the maxim *keep all objectives satisfied* and can be ensured by providing implementation details for leaf propositions which are of type behavior or constraint. Assumption propositions are to be trusted. Motivation propositions cannot be shown as leaves in a refined tree.

[1] ArchiMate® is a registered trademark of The Open Group.

TABLE I. TYPES OF PROPOSITION AND THEIR NOTATION

Proposition type	***Notation***
Motivation	/m/
Behavior	/b/
Constraint	/c/
Assumption	/a/

TABLE II. REFINEMENT RULES FOR GOAL SKETCHING

Proposition type	***Refinement***
Motivation	/m/ | /b/ /a/ | /c/ /a/
Behavior	/b/ {/a/}
Constraint	/c/ {/a/}
Assumption	/a/

Figure 1 shows our KAOS-like goal sketching notation. The interpretation is B $\wedge$ C $\wedge$ A' $\models$ A where A, B, C and A' are goals. A' is included for completeness and represents any propositions that are in A that are not satisfied by B and C together; this includes compositional relationships of all propositions in A. We use the term *requisite* to refer to propositions which are of type behavior, constraint or assumption. Once we have refined motivation goals to requisite goals the requisites can be refined using the refinement rules in Table II. Motivation goals can be refined into requisites on the plateau-to-be (i.e. /c/ and /b/ propositions) provided that they involve a justification in the form of an assumption /a/ (see Fig. 2 where P $\wedge$ Q $\wedge$ R $\models$ C). P is the assumption (/a/) that justifies the entailment (akin to *Justifications* in Problem Oriented Engineering [10]).

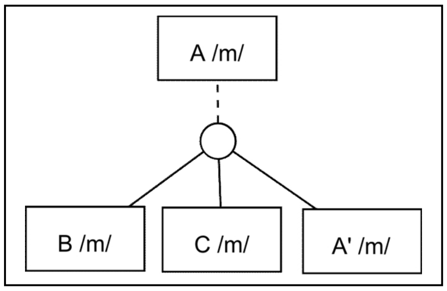


Fig. 1. Refinement of /m/ goal using our goal sketching notation

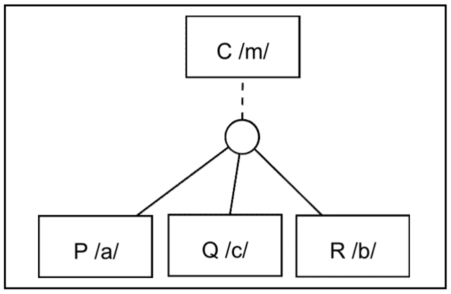


Fig. 2. Refinement of /m/ goal into requisites

In Figure 3 we see a refinement of a behavior proposition.

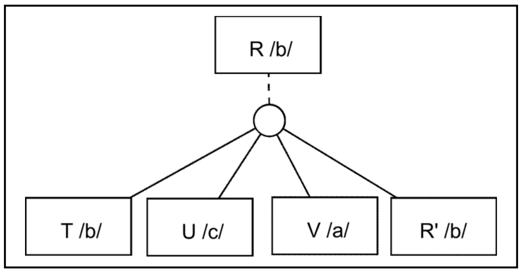


Fig. 3. Refinement of /b/ goal using goal sketching notation

Once the requisites have been sufficiently refined they can be assigned to system elements of the system-to-be for operationalization.

III. ENTERPRISE ARCHITECTURE MOTIVATION MODELING

Enterprise architecture is concerned with taking a holistic view of a company, its structure and its processes. ArchiMate is an open and independent modeling language for enterprise architecture that is supported by a number of different tool vendors and consultancy firms. It is maintained by The Open Group ArchiMate Forum which has more than 80 member organizations. The ArchiMate certification program has certified over 2,200 individuals world-wide and there are 9 certified ArchiMate tools available.

Typically a motivation model has stakeholders and drivers that can be assessed to produce goals (which are the desired ends). A *driver* is a kind of generalized concern such as *customer satisfaction*.

Figure 4 illustrates the ArchiMate motivation model elements and structural associations. It ignores *influences* associations which whilst very valuable are beyond the scope of this paper. In the interests of simplicity it also ignores the possibility of many-to-many associations.

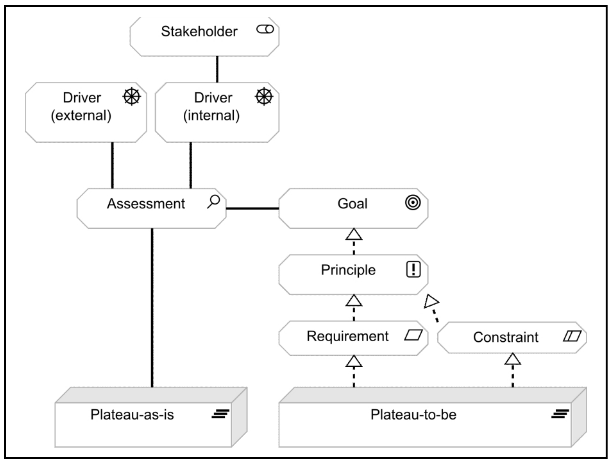


Fig. 4. ArchiMate Enterprise Architecture Motivation Model Elements

Figure 4 shows a system architecture in existence called the plateau-as-is. There is an assessment of that state based on one or more *drivers*. Drivers create, motivate, and fuel change in an organization [11]. Two drivers are indicated (one internal and one external). The *assessment* is the outcome of an analysis of one or more drivers [11]. It contains goals for an improved future state of the plateau, which we call the plateau-to-be. A goal is satisfied (realized) through a realization based on some identified *principle* of design: "a normative property of all systems in a given context, or the way in which they are realized" [11]. Realization is shown in Fig. 4 by an arrow with a dashed line that points from means to ends.

A principle is realized, as shown in Fig. 4, by requisites (requirements and constraints) being placed on the elements of the plateau-to-be. These elements are taken from the ArchiMate three layer model (the Business, Application and Technical Infrastructure models) [11].

IV. Synthesis of Goal Sketching and Motivational Modeling

In ArchiMate "a goal is defined as an end state that a stakeholder intends to achieve … or a produced value" [11]. This corresponds to a motivation goal (/m/) in goal sketching.

A behavioral goal in Goal Sketching is equivalent to the IEEE concepts of Capability and Condition [17] which in turn correspond to requirements in the ArchiMate language: "a requirement is defined as a statement of need that must be realized by a system… requirements represent the "means" to realize goals" [11]. A constraint is a restriction on the way in which a system is operationalized [11], [1] and [16].

A. Refinement of Motivation Goals

ArchiMate Goals can be decomposed as illustrated in Fig. 5 which is the equivalent of Fig. 1.

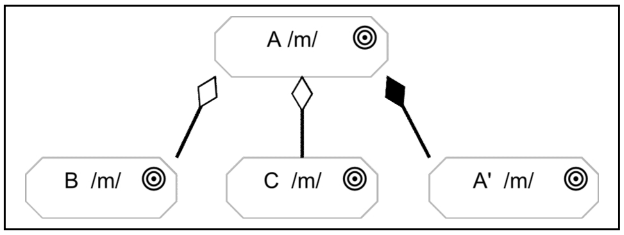


Fig. 5. Refinement of /m/ goal using ArchiMate Motivation Elements

In the ArchiMate language motivation goals are indicated with a target symbol. The refinement is effected using the combined concepts of aggregation (shown by a hollow diamond) and composition (shown by a black diamond) with the usual semantics that aggregation indicates that an object groups a number of other objects whereas composition indicates that an object consists of a number of other objects. The presence of the composition element (eg A') assures semantic entailment.

B. Refinement into Requisites

In the ArchiMate language refinement into requisites is done through the use of a *principle* element which we take to be an assumption (/a/). Figure 6 shows this: C, P, Q and R from Fig. 2 become C, Principle, Q and R.

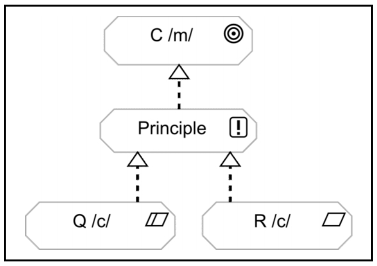


Fig. 6. Refinement of /m/ goal into requisites using ArchiMate notation

In Goal Sketching we do not assume that the transition from a motivation goal to requisites can always be normative; if the refinement is entirely new to those involved, it is classed as radical. In contrast, normal analysis (which is equivalent to a safe *divide and conquer* approach) involves dividing a problem into parts and repeating this process until each part is simple enough to be solved in its entirety. Thus our normal and radical types of analysis are analogous to the normal and radical types of design proposed by Vincenti [9]. This idea allows us to modify the definition of principle to: *the realisation of a goal which may involve either a radical proposition or a normative proposition*.

C. Further Refinement of Requisites

Requisites may be refined until there is sufficient clarity to place responsibility for their realization on specific elements of the plateau-to-be. Figure 7 equivalent of Fig. 3. It shows our use of assumptions.

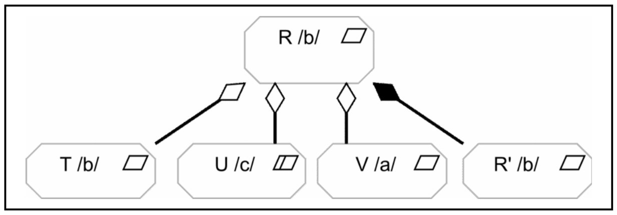


Fig. 7. Refinement of /b/ goal using ArchiMate Motivation Elements

D. An Approach to Motivation Modeling

The enterprise architecture stakeholder, driver and assessment elements provide the structure to summarise our investigations about stakeholders and their concerns. The assessments establish the headline objectives for a desired new plateau. The details of this process are outside the scope of this paper.

The superordinate goal statement is a motivation goal and as described above can be refined until at some point a principle is introduced as an assumption to define the realization approach (as in Fig. 6). Realization progresses by refining the requisites to the point that they can assigned with confidence to elements of the plateau-to-be.

E. Operationalization

The plateau-to-be comprises a connected set of the ArchiMate core model elements (i.e. Infrastructure Layer, Application Layer and Business Layer). Any of these elements can be assigned responsibility to realize the desired requisites exposed at the leaves (see Fig. 8).

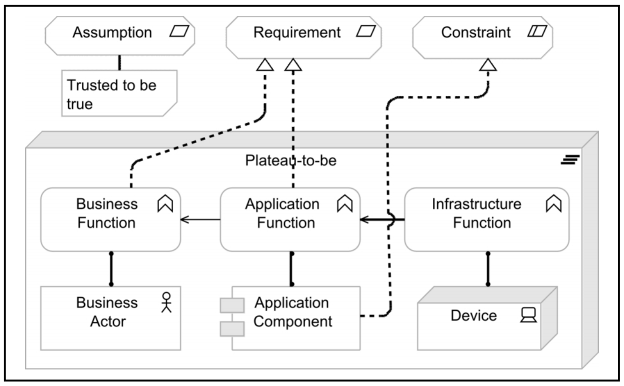


Fig. 8. Operationalization of requisites

The assumption is to be trusted (it is not imposed by the engineered parts of the plateau-to-be). Lazy refinement means that we can satisfy the requirement by the combined effects of the business process and the function of the application component [13]. The constraint is satisfied in the way the application component is constructed.

V. Example

This example shows the use of assumptions and the role of principles when a non-normative problem is addressed. The example (RouteMaster) concerns the fictional University of Loamington. It is a large campus university with many buildings with paths. The Library has a central building but half its stock is in shelves distributed in buildings across the campus.

The library has a library management application component and access to it is available across the university local area network. The librarian concerned with the efficacy and efficiency of the library service to users has commissioned an assessment the main features of which are reported in Fig. 9.

Using the ArchiMate motivation model we capture this information as shown in Fig. 10. This shows stakeholders, drivers and the inclusion of the assessment referring to the plateau-as-is and leading to goals (as yet TBD) for the desired plateau-to-be.

The driver elements are efficacy and efficiency. The assessment (Fig. 9) refers to the condition of the plateau-as-is and establishes goals for a desired plateau-to-be, which are to be achieved using our Goal Sketching approach to the Motivation Model. Figure 11 shows the plateau-as-is.

The shelves on which items of library stock are kept are located in various buildings distributed over the campus. This frustrates library users (a weakness in terms of efficacy of service) and causes staff to be distracted by a demand to explain the best route to an item's shelf. The frustration is greater when the library user is working at a network node that is not close to a member of library staff. We need to improve the users' access to library stock by providing a self-help service to show the shortest route between the network node (where the user is accessing the library catalogue) and the shelf (on which the item is located). This self-help service should save staff time though there may be a need to maintain the data that represents routes on the campus. We do not have access to a normative exemplar for this problem and so the problem is radical and the best we can do is create a pilot implementation that can subsequently be refined.

Fig. 9. The assessment provided to the librarian

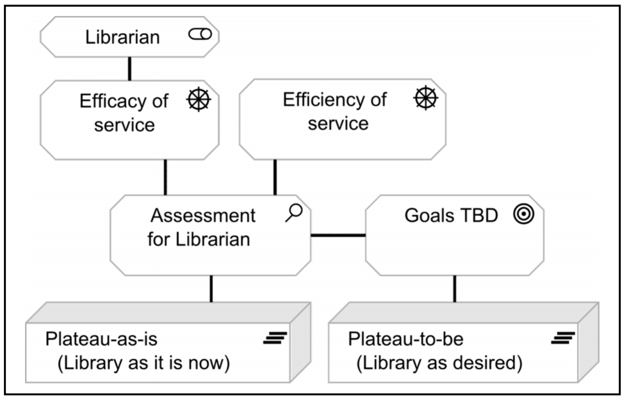
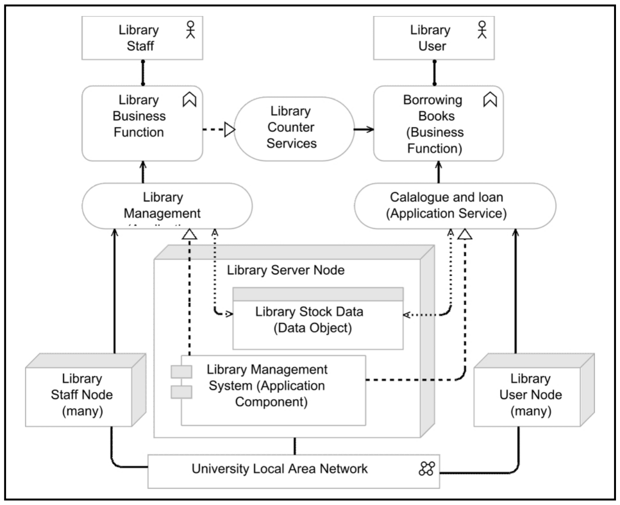


Fig. 10. RouteMaster Specialization of Fig. 9

Figure 11 shows the library staff and library user computing nodes (terminals) connected via the university local area network to the library server node. This node supports the library management system application component and the working data. The application component's functionality exposes library management services, used by the library staff conducting their library business functions and by the library users performing their business functions of looking up and borrowing books.



Library Staff
Library User
Library Business Function
Library Counter Services
Borrowing Books (Business Function)
Library Management
Calalogue and loan (Application Service)
Library Server Node
Library Stock Data (Data Object)
Library Management System (Application Component)
Library Staff Node (many)
Library User Node (many)
University Local Area Network

Fig. 11. UoL Architecture Plateau-as-is (simplified)

An additional business service provided by the staff in conducting their library business functions is the counter service which is used by library users in their Borrowing function

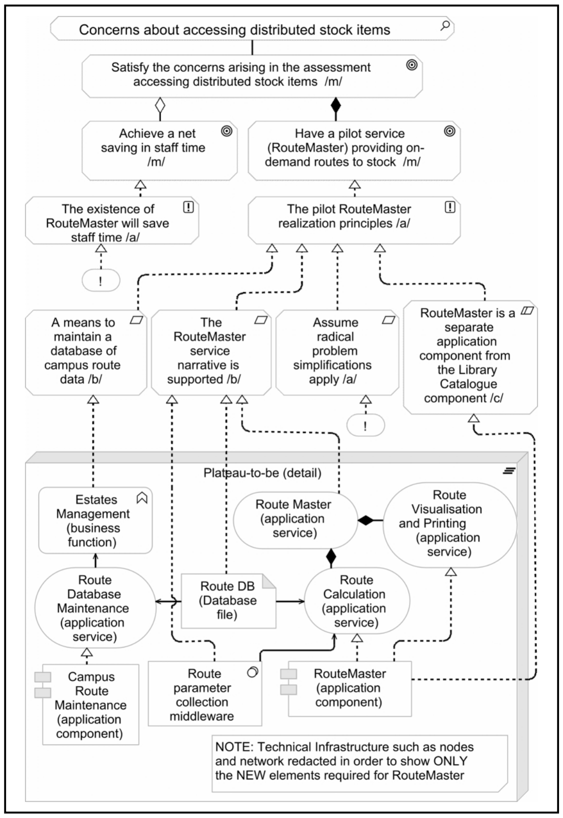
The domain context of our requirements analysis is shown by the business layer activity, the presence and use of application components and the presence and use of technical infrastructure.

The *goals TBD* in Fig. 10 begin with a headline motivation goal associated with the assessment "Satisfy the concerns arising in the assessment accessing distributed stock items". This refines into two sub-goals "Achieve a net saving in staff time" and "Have a pilot service (RouteMaster) providing on-demand routes to stock". As a principle we assume that the existence of the RouteMaster will save time. This is an assumption and so can only be trusted.

The key to a structurally complete analysis is the principle "The pilot RouteMaster...". The documentation for this is summarised in Fig. 12. Figure 13 shows the goal "Have a pilot service…" refined and operationalised with new elements of the plateau-to-be. The operationalization of the leaf goals (the requisites "A means…", "The RouteMaster Narrative..." etc.) is shown assigned to new elements of the plateau-to-be. Only enough of the plateau-to-be is shown to reveal the operationalization. The equivalent representation in Goal Sketching notation is shown in Fig. 14.

RouteMaster Service Narrative: a database of campus route data (Route DB) must be maintained as a means of assembling the route parameters. These data can then be used in a route solver algorithm. The results should be provided on screen to the user with an option to print on a convenient network printer. Route determining parameters include: the starting node (where the user is working), the catalogue information about the item's shelf and availability, and the user's preferences concerning disabilities. This service must work alongside the existing library management system with minimum coupling to it; hence it should be constructed as a separate application component. As this is a pilot to a radical problem it is acceptable to simplify the service to ignore user preferences and calculate the shortest path.

Fig. 12. Documentation for the Principle Element



Concerns about accessing distributed stock items
Satisfy the concerns arising in the assessment accessing distributed stock items /m/
Achieve a net saving in staff time /m/
Have a pilot service (RouteMaster) providing on-demand routes to stock /m/
The existence of RouteMaster will save staff time /a/
The pilot RouteMaster realization principles /a/
!
A means to maintain a database of campus route data /b/
The RouteMaster service narrative is supported /b/
Assume radical problem simplifications apply /a/
RouteMaster is a separate application component from the Library Catalogue component /c/
!
Plateau-to-be (detail)
Estates Management (business function)
Route Master (application service)
Route Visualisation and Printing (application service)
Route Database Maintenance (application service)
Route DB (Database file)
Route Calculation (application service)
Campus Route Maintenance (application component)
Route parameter collection middleware
RouteMaster (application component)
NOTE: Technical Infrastructure such as nodes and network redacted in order to show ONLY the NEW elements required for RouteMaster

Fig. 13. Goal Refinement and Operationalization for RouteMaster

It is evident from figures 13 and 14 that the scope and assumptions guiding the RouteMaster service have been established. The leaf goals can be now refined further to reveal detailed acceptance criteria.

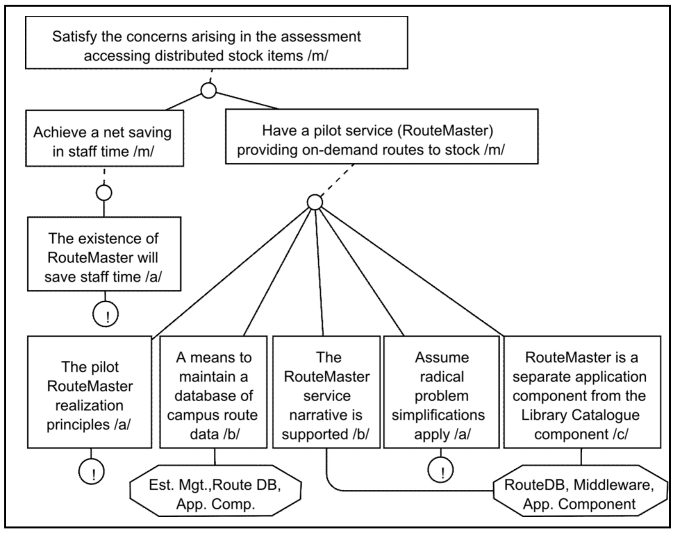

Satisfy the concerns arising in the assessment accessing distributed stock items /m/
Achieve a net saving in staff time /m/
Have a pilot service (RouteMaster) providing on-demand routes to stock /m/
The existence of RouteMaster will save staff time /a/
!
The pilot RouteMaster realization principles /a/
A means to maintain a database of campus route data /b/
The RouteMaster service narrative is supported /b/
Assume radical problem simplifications apply /a/
RouteMaster is a separate application component from the Library Catalogue component /c/
!
!
Est. Mgt.,Route DB, App. Comp.
RouteDB, Middleware, App. Component

Fig. 14. Goal Sketching equivalent of Fig. 13

VI. Related Work

Goal-oriented requirements engineering has been much researched since its inception [13], [14] and forms the basis for our work [2], [3], [12], [15]. We contend that whilst extremely useful goal-oriented requirements engineering has not yet been widely adopted in industry, possibly because of its formality. Goal sketching offers a lightweight approach to goal-oriented requirements engineering which we hope has fewer barriers to adoption. Goal sketching shares the maxim *keep all objectives satisfied* with KAOS [4] but relies only on semantic entailment. Instead of using the Object, Operation and Responsibility models of KAOS, goal sketching allows the user to adopt any domain modelling method to find the elements that will ultimately operationalise the goals such as context diagrams from problem frames [16], or core models from the architectural plateaux of ArchiMate, which we concentrate on here. The *i** modelling framework [17] allows goal refinement (in a similar way to goal sketching) and also provides a motivational aspect but is relatively complicated for use in industry where time is of the essence.

Early work on the use of graphical notations for requirements modeling includes that of Heaven and Finkelstein [18], which shows how UML models could be used instead of the KAOS object model. Our work has an advantage over this as far as requirements are concerned because ArchiMate focuses on architecture rather than design.

Software engineering has been greatly influenced by work on design and analysis patterns [19] and this influence also applies to our work, especially our work on goal refinement.

One of our objectives is to ensure that a holistic worldview of a system-to-be is maintained throughout analysis. On similar lines, an architecture-based approach to software development is discussed by Hall et al. through an extension to problem frames [20]. The reconciliation of formal and nonformal descriptions is also considered by Hall, Rapanotti, and Jackson in their work on Problem Oriented Software Engineering [6].

More recently, Engelsman and Wieringa have reported on the relationship between enterprise architecture and goal-oriented requirements engineering [21]. The authors observed that their initial language for enterprise architects was regarded as too complex to be practical.

VII. Discussion and Further Work

We have juxtaposed Goal Sketching with the ArchiMate Motivation model. In doing this we have shown how the motivation model's goal, principle and requisite elements can be used in Goal Sketching. This affords a degree of corroboration of the Goal Sketching proposition types (/m/, /b/, /c/ and /a/). With some limitations Goal Sketching provides an intuitive approach to the construction of full scope enterprise migration and so also to system development requirements.

We have proposed an extension of requisites to include assumptions. We have also cast the ArchiMate principle type element as an assumption. By these devices we have raised assumptions to become a first class part of the analysis as advocated by Lehman [7]. We have also shown how the principle element records design debate. In contrast to the usual ArchiMate usage we have shown its use on a problem that is radical.

Goal Sketching includes the possibility of alternative (OR) refinement arguments. Although beyond the scope of this paper alternatives are possible in the Goal Sketching extension of motivation modeling. Key tactics include use of goal specialization relationships at a motivation level and multiple principles at a realisation level. A rigorous treatment of these ideas is required. This work will also explore the formality of load-bearing assumptions [8].

Continuous stepwise refinement is enacted through an ever unfolding story of breadth before depth [19]. Stopping time occurs when we have exposed enough acceptance criteria on system elements to allow progress to implementation. This approach may be sufficient alone but equally it can be used with the UML, problem frames, user stories, and use case analyses among others. We will continue to validate this work and improve its agility and ease of use.

VIII. Conclusions

In this paper we have discussed a method for requirements modeling. The method is built using the synergies between goal sketching, (our lightweight approach to goal oriented requirements engineering) and the ArchiMate enterprise architecture modeling language. We propose the inclusion of assumptions as first class entities in the ArchiMate modeling language and the extension of *principles* from the language to allow radical as well as normative analyses. We have demonstrated the usefulness of the method using an example of a simple university library system.

REFERENCES

[1] K. Boness, R. Harrison, “Goal sketching and the Business Case”, The Fourth International Conference on Software Engineering Advances (ICSEA 09), Portugal, October 2009 (IARIA).

[2] K. Boness, A. Finkelstein, R. Harrison, “A lightweight technique for assessing risks in requirements analysis”, IET Software, 2(1), pp 46-57, Feb. 2008.

[3] K. Boness, A. Finkelstein, R. Harrison, “A method for assessing confidence in requirements analysis”, Information and Software Technology, 53, pp. 1084-1096, 2011.

[4] A. Dardenne, A. Van Lamsweerde, and S. Fickas. "Goal-directed requirements acquisition." Science of computer programming 20.1 1993, pp. 3-50.

[5] IEEE Std 1233-1998. Guide for Developing System Requirements Specifications. The Institute of Electrical and Electronics Engineers, Inc. IEEE Software Engineering Standards Collection. 1999.

[6] J. G. Hall, L. Rapanotti, and M. Jackson. "Problem Oriented Software Engineering: A design-theoretic framework for software engineering." Software Engineering and Formal Methods, 2007. SEFM 2007. Fifth IEEE International Conference on. IEEE, 2007.

[7] M. M. Lehman, "The role and impact of assumptions in software development, maintenance and evolution." Software Evolvability, 2005. IEEE Int. Workshop on. IEEE, 2005.

[8] J. A. Dewar, Assumption-based planning: A tool for reducing avoidable surprises. Cambridge University Press, 2002.

[9] W. G. Vincenti, What engineers know and how they know it. Baltimore: The John Hopkins University Press, 1990.

[10] J. G. Hall, L. Rapanotti, and M. Jackson. "Problem oriented software engineering: Solving the package router control problem." Software Engineering, IEEE Transactions on 34.2 2008, pp. 226-241.

[11] The Open Group, ArchiMate 2.1 Specification – Technical Standard, The Open Group, 2013.

[12] K. Boness, R. Harrison, K. Liu, Goal sketching: An Agile Approach to Clarifying Requirements, IARIA, 1(1), Jan 2009, pp. 1 – 13.

[13] A. Van Lamsweerde, "Goal-oriented requirements engineering: A guided tour." Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on. IEEE, 2001.

[14] A. Van Lamsweerde, "Requirements engineering: from system goals to UML models to software specifications." 2009.

[15] K. Boness, R. Harrison, Goal Sketching from a Concise Business Case, IARIA, 3(1), Sep 2010, pp. 90 – 99.

[16] M. Jackson. Software requirements and specifications. Reading: Addison-Wesley, 1995.

[17] E. S. K. Yu, "Towards modelling and reasoning support for early-phase requirements engineering." In Requirements Engineering, 1997. Proceedings of the Third IEEE International Symposium on, pp. 226-235. IEEE, 1997.

[18] W. Heaven and A. Finkelstein. "UML profile to support requirements engineering with KAOS." IEE Proceedings-Software 151.1 (2004): 10-27.

[19] S Adolph, A Cockburn, and P Bramble. Patterns for effective use cases. Addison-Wesley Longman Publishing Co., Inc., 2002.

[20] J. G. Hall, M. Jackson, R. C. Laney, B. Nuseibeh, and L. Rapanotti. "Relating software requirements and architectures using problem frames." In Requirements Engineering, 2002. Proceedings. IEEE Joint International Conference on, pp. 137-144. IEEE, 2002.

[21] W. Engelsman, R. Wieringa, Goal-Oriented Requirements Engineering and Enterprise Architecture: Two Case Studies and Some Lessons Learned. In: Regnell, B., Damian, D. (eds.) REFSQ 2011. LNCS, vol. 7195, pp. 306–320. Springer, Heidelberg 2012.