A MODULARIZED APPROACH TO THE INTEGRATED ENVIRONMENT

Faraj I. and Alshawi M.

Department of Surveying , University of Salford
Salford M5 4WT, United Kingdom

ABSTRACT: *The lack of a high level structure for and a full understanding of an integrate environment has led to the development of a number of small integrated applications in various fields of construction. This paper has addressed this issue and proposes a strategic, but generic, framework for an Integrated Construction Environment (ICE). The framework is based on three main components; the central core data models, the construction applications, and project specific information. The paper has adopted IDEF0 notations to represent a process based model which shows the various activities involved in the ICE along with their relationship with each other. The framework highlights a methodology for operating such an environment addressing specifically the three main issues outlined above*

KEYWORDS: *Integrated construction environment, process modelling, product models, structured analysis, IDEF0*

INTRODUCTION

The concept of computer integrated construction proposes that if a range of different applications can be integrated with each other then cumulative benefits would be gained throughout the design and construction processes. It has been widely recognised that in order to achieve the best performance, such integrated systems need to share data via a central core, i.e. a data model. Great efforts have, therefore, been devoted towards the development of a flexible and comprehensive data structure, for a central data base, with the aim of serving all possible downstream applications. However, the general concept and structure for the overall environment has not been fully addressed.

The complexity and the vast amounts of information involved in any construction project, and the lack of standards have made the process of producing an integrated environment very difficult. On the other hand, the current systems used by industry are fragmented and can no longer serve the rapidly changing business demands. A study in the field of design and construction integration (Yamazaki, 1992), has identified the following as being the major problems in the current systems:

a) Basic information and knowledge about construction technologies is not shared between designers and contractors.

b) Interactive procedures, to apply construction knowledge at early design stage, have not yet been developed.

c) No systematic evaluation and feedback of relevant data and information from construction sites.

An integrated environment can address such problems whereby all possible construction applications can be integrated under one environment. Such an environment should be developed where extension to the domain of applications that it can support must be considered. In the development of such environment which is capable of supporting many domains of construction, care must be given to a number of issues such as data redundancy, integrity of data, conflict of constraints, multiple views of data, data management, and modifiability. If this is developed and implemented effectively, it could assist in reducing lead times reduce cost, and increase project quality.

Our experience tell us that if such an integrated environment is to be developed the issue of integration must be considered and given a priority at a very early stage of the development rather than development of specific knowledge to serve particular domains in isolation and then bring them together. This will increase the likelihood of encountering the problem mentioned above, namely, data redundancy, integrity of data, conflict of constraints, and data management.

The complexity of the construction applications and the vast amount of information involved in a project have hindered the development of such an environment. Moreover, the lack of a high level structure (a strategic vision) of such an environment, i.e. high level processes along with their data models, and their relationships with each other and with external applications, has led to the development of a number of small integrated applications in various fields of construction. Therefore, to enable such an environment to be established, a developer's view need to be created to clearly define its requirements and to present the high level interactions between the various activities involved in the integration process. This paper addresses the issue of integration and proposes a strategic, but generic, framework for establishing an Integrated Construction Environment (ICE). The first section defines the terminologies within the context of the proposed framework. Then a conceptual presentation of the framework is discussed followed by a brief explanation of the technique used to model the proposed framework. A full process analysis of the various activities required to successfully establish such an environment is then explained. Any details such as product models' structures, applications' features, etc. are outside the scope of this paper. In this paper we are not presenting novel ideas, but we are reporting on our experience and findings we came across during the implementation of SPACE

DEFINITIONS

Project: is used to refer to a concept that can be constructed using construction processes, it is not a physical object.

Product Data Model: is a piece of software that describes the form and content of the product model.

product Model: is a software representation of project type that support a project throughout its life cycle from design, construction to the maintenance of the project. Product model is a populated (instance of the) product data model .

Integration: the bringing together of engineering applications and data so that they operate harmoniously together (McKay and Bloor, 1991).

Application: is a piece of software that perform a specific task on the project during its life cycle.

APPROACHES TO INTEGRATION

A number of researchers have identified that a central database is one of the keys to integration. Others argue that the implementation of such a database is impractical and proposes a central data base is better achieved as a collection of smaller, more manageable databases. In either case a data model must be devised which is capable of supporting the different view of the applications. Each application has its own view on the central data model.

One approach to realise such concepts is to have a group of researchers working on specialised topics and then join the modules of the individuals later. This approach has a number of disadvantages, individuals modules don't always fit together in a coherent and tidy manner and may require major amendments. Data duplications and conflicting constraints is common with this approach due to applications own view of the data. For example, a room may be defined as a four walls, a ceiling and a floor. A construction planning application may need to know about the dimensions and the position of each wall, where as the bill of quantity needs to about the number of wall and cost. We could conclude from this approach that information about each application must be determined before integration can begin.

The approach we have adopted enables individuals to work separately to develop data models for specific applications. A framework is developed which consists of slots that enables individual data models to be slotted is defined. The advantage of this framework is that it enables the individual data models to be slotted and tested at a very early stage. Conflicts and inconsistencies can be identified and addressed before the data structure is finalised.

The Architecture of the Proposed Environment

The architecture presented in Figure 1 aims at integrating various construction applications under one integrated construction environment (ICE). It consists of three main parts. These are: ICE data models, construction applications and the user interface.

The ICE data models consist of two parts, the project data ad applications specific data models. The project data models provides a structure where all data needed by the downstream applications is defined. Through this structure each applications knows what is the type of data (e.g. integer, list, user defined, etc), where to obtain it

from and where to write the newly generated data if required. Without the data models, management of information within the integrated environment becomes extremely difficult if not impossible.

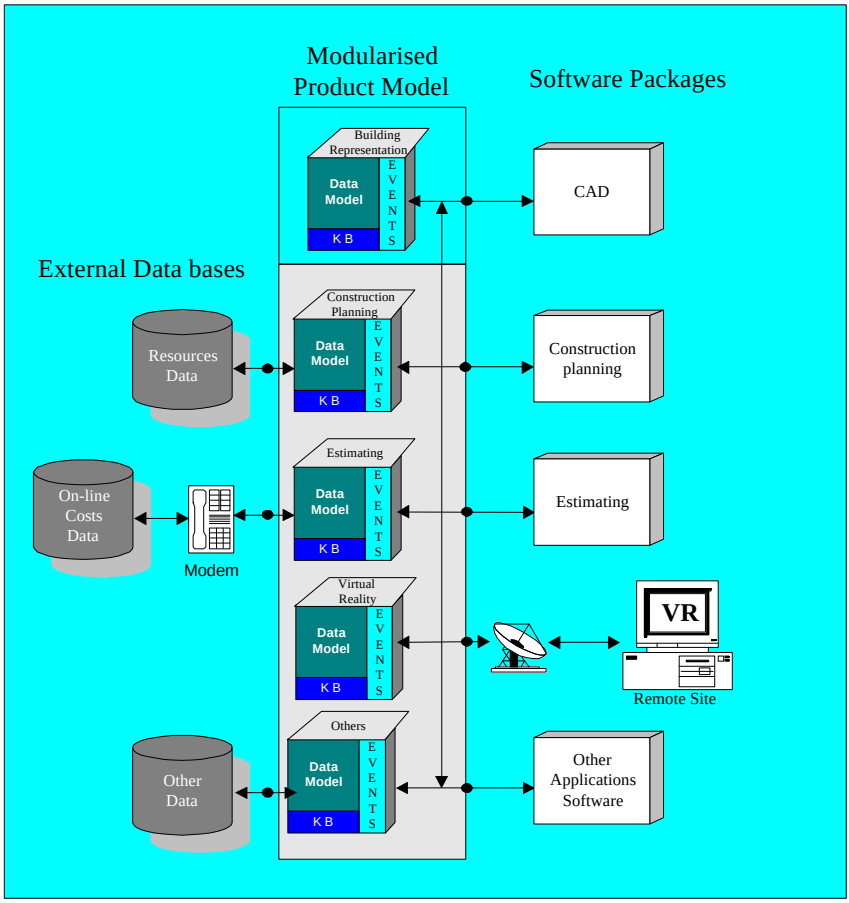

Figure 1: The proposed framework for the ICE

One of the characteristic of the project data model is its ability to support the life cycle of the project from concept through design and construction to decommissioning. The representation of project life-cycle is not an easy task and may involve a number of experts working on different areas of the life-cycle. for example, architects, construction planners and site layout planners. Therefore it would be of value to segment the data to a number of models each is concerned with a particular stage of the project life-cycle. This enables individual experts to work separately and ease the data management of environment, however, the framework of the project data model will enable these data models to be tested at any stage of the development to highlights problems of inconsistencies and data duplication during the development stage.

A mechanism must exist that would enable the project model to request data from the specific models and vice-versa. This will be discussed further in section (the central core). The mechanism may take many forms depending on the tools of implementation.

Each application specific model provides a structure which contain all the data and relationships between data to perform the task required by the application it is serving. This data can be shared by other data models. Construction applications interfacing with the data models obtains data from the instances of these data models i.e. once there is enough information to enable these applications to perform the required task. Application can be part of the database they are referred to as "internal", or they may exist outside and interfaced to it. How these applications are interfaced is still a research issue.

A user interface to the ICE will assist in populating the data models with data knowledge, constraints associated with a particular project. The interface enable the user to interact with both the project and applications specific models and the applications.

Information Flow Within The ICE

To convert a project from a concept t a physical object a number of processes need to be carried out. These processes form a subset of the life-cycle. The other parts of the life-cycle may include usage and decommissioning which are not going to be discussed any further.

In order for the concept (may be initiated by a client requirements) to be realised, a design may be produced which should satisfy the specifications conveyed to the designer by a client. In ICE it is likely that this design is generated by a design system (e.g. drafting, solid modeller, etc.) where large amount of projects' information can be extracted (Dym and Levitt 1991). Applications could interrogate the design database to obtain data required to enable these applications to perform their task.

Geometric information is one of the most common data which shared between a large number of ICE applications. Most CAD systems provide data in a form that is not suitable for ICE application purposes such as lines, points. This data need to be translated to a higher form know here as building elements e.g. walls, doors, etc. The translated data form part of the project data model framework in the realisation part (see later). At any stage of the design users should be able to invoke applications to assess the project at this stage. For example, the cost of the so far developed design can be dynamically determined by running the estimating application. The central core, in this instance, transfers the design elements along with their specifications and quantities to the estimating package to produce either the total cost or cost break down of the current design product. Users should be able to alter the generated cost figures, add cost constraints on certain design elements, etc. using the estimating package interface. This altered information is then transferred to the central core where actions are triggered-off if the altered information is non-feasible or does not comply with regulations, standards, in-house database, etc. The application generated data can be stored if needed in the project model. For example, the construction plan generated by

the construction planning applications can be stored in the project model as it forms part of the project life-cycle. This then can be checked against the specifications to establish the conformance of the result with that of the specifications.

The Central Core

At the heart of the ICE a number of data models may exist which support the applications to perform their functions during the project life-cycle.

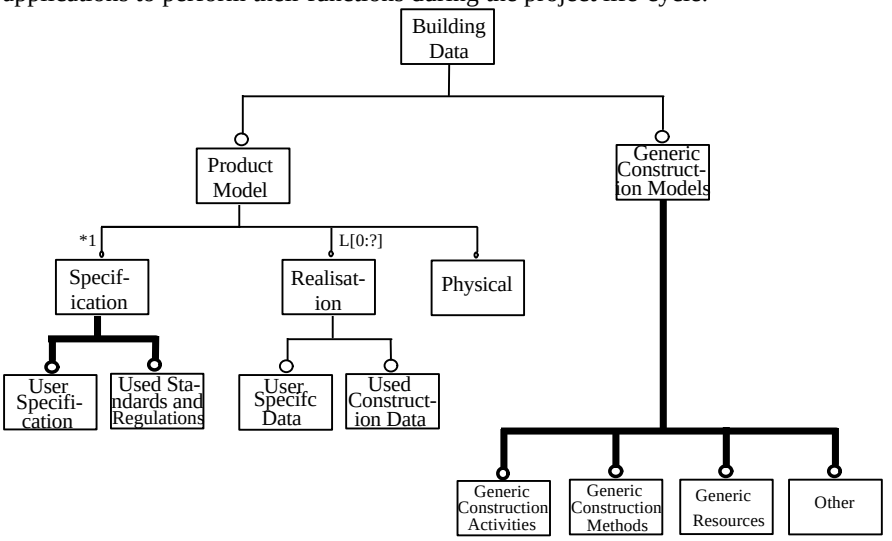

Figure 2: Typical Models of the ICE

The central core consists of a number of data models which describes the life cycle of the project type under consideration. These models have been classified as product data model and other data models, Figure 1. The extent and structure of the product data model depend on the scope, and the main objectives of the ICE. Different ICEs may have different structures of product data models, level of abstractions, different coverage of the project's life cycle, etc. Figure 2 shows an Express-G representation of a typical product data model covering the life cycle of a building, i.e. building elements at various levels of abstractions. Three main parts are involved in this product data model; Specifications, Realisation, and Physical. These parts produce a complete definition of a class (object) through its life cycle e.g. the Specification part contains data referring to the specification of the element (a project must have one specification). The Realisation part covers all the necessary data required to construct it (i.e. how the specifications are to be realised), an element can be realised in different ways, for example many companies can satisfy the specification . The Physical part holds geometric and other data that describe the physical status of the element.

Other data models represent data required by the various applications within the ICE such as Construction Planning Data Model, Site Planning Data Model, Specification Data Model, Estimating Data Model, etc. Each data model should be developed to

achieve a well defined set of objectives and to fulfil the need of a particular application. For example, a construction planning application requires a data model to support the information required by the application e.g. generic construction activities, resources available, construction methods, etc.

Information between the product data model and other data models may overlap and therefore the concept of data sharing must be applied. For example, Figure 2 shows that part of the specification defined for a particular element can be similar to that of the Specification Model. In such a case, similar data should be shared between the various data models using the data sharing principles. This is an import issue which has to be addressed when designing and implementing such models.

STRUCTURED ANALYSIS AND DESIGN TECHNIQUE (SADT)

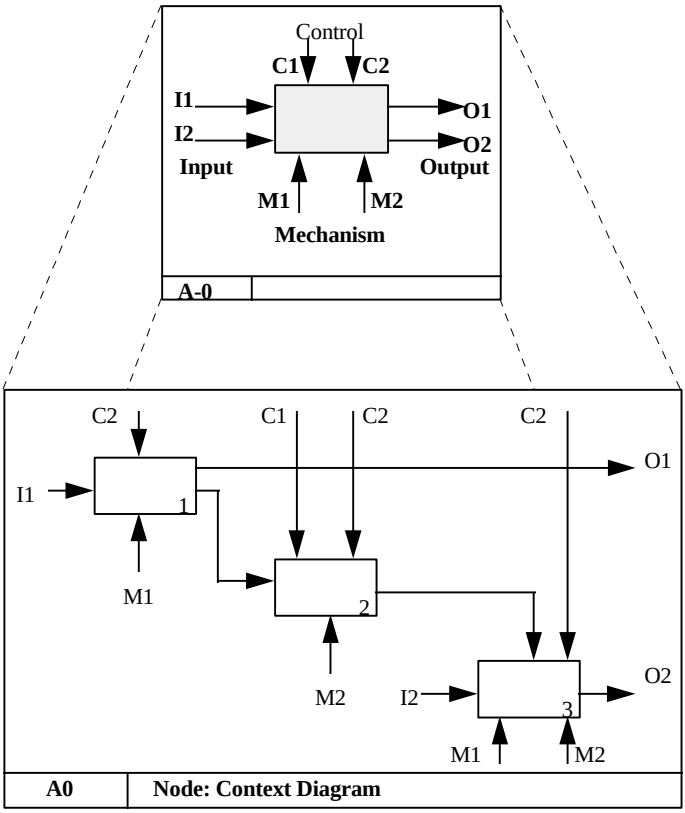

Figure 3: The Context Diagram and its decomposition (Marca and McGawn, 1988)

Structured Analysis and Design Technique, SADT has been used to represent the proposed strategic framework for an Integrated Construction Environment (ICE), where design and other downstream applications can be integrated. SADT was originally developed by Ross in 1977 (Ross, 1977) (Ross and Schoman, 1977). Since its emergence it has been widely used in the development of Computer Aided

Engineering (CAE) applications. The ICAM, US Air Force Integrated Computer Aided Manufacturing, adopted the technique and resulted in the IDEF0, ICAM Definition Method Zero, which is now been used by the STEP initiative to describe activity based models. An important feature of the SADT is its time dependency i.e. activities are sequenced according to their time of execution. This criteria makes SADT differ from other structured techniques such as Data Flow Diagrams.

SADT is an activity based modelling technique which adopts top-down approach whereby high level activities are decomposed into low level activity exposing more and more details of the system being analysed. The central feature of SADT is the "SA box", Structured Analysis box, and its concept of "decomposition" (Marca and McGawn, 1988). The SA box is a graphical representation of an activity along with its main four types of data flows; the main activity's input, the activity's constraints and controls, tools required to perform the activity, and the activity's main outputs. The graphical notations for a SA box are as follows (Figure 3):

a. The activity is presented as a square, normally rounded, with the activity's name is written at the centre.
b. The activity's main input is presented as a headed arrow pointing to the left side of the activity box with the input description written on it.
c. The activity's main output is presented by a headed arrow pointing away from the right side of the activity box with the output description written on it.
d. The controls and constraints which are imposed on the activity are again donated by arrows pointing to the top of the activity box.
e. The tools required to perform the activity, mechanism, are shown by arrows pointing toward the bottom of the activity box.

SADT Structure Method

The top level activity diagram of a system, as represented by the SADT, is called the Context Diagram, Figure 4. The Context Diagram clearly describes the purpose of the system being investigated i.e. as represented by its top level activity. At a high level of abstraction, this activity shows the data input required by the system, the data output generated by the system, the tools required to perform the activity, and the constraints within which the activity has to operate. The Context Diagram is decomposed further into several low level diagrams by decomposing the main activity into its constituents sub-activities. This implies that the activity described by the context diagram is performed by the sum of its sub-activities. Figure 5 shows the hierarchical structure of such subdivision.

The SADT graphical structure consists of SA boxes with arrows linking the various involved activities, as shown in Figure 1. A diagram should have between 3 and 6 SA boxes, each representing a single activity.

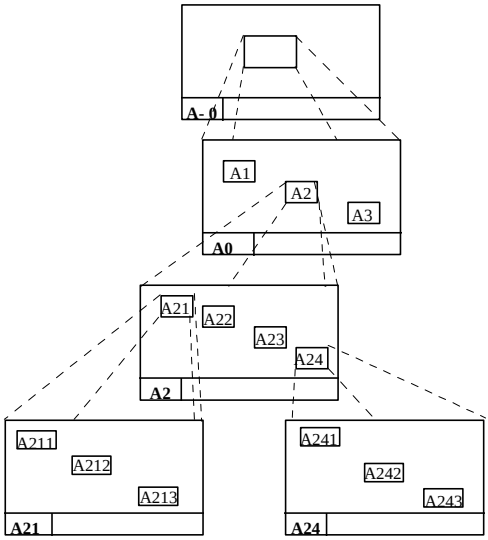

Figure 4: The hierarchical structure of SADT diagram (Marca and McGawn, 1988)

Information shown on the diagram represents constraints, relationships and/or data flows between activities. They are grouped into four categories, as previously explained. Input, Control, Output, and Mechanism (ICOM). The input of an activity is transformed to an output by executing the activity. Controls and constraints ensure that transformation is only applied under the appropriate circumstances while a mechanism defines how an activity can be performed. Thus, an activity can be fully specified, including its interface with other activities, by a combination of input, control, output and mechanism. Furthermore, arrows entering or leaving an activity must always correspond to those at higher level diagrams.

PROCESS MODELLING FOR THE PROPOSED ICE

The proposed approach discussed in the paper represents a developer's view of an Integrated Construction Environment. It is a representation of generic activities along with their relationships which demonstrate how downstream applications can be integrated with the central core data models. The process modelling is represented in three layers, starting from the Context Diagram down to level 1 process decomposition. The following sections explain the proposed approach using the IDEF0 notations.

The Context Diagram (A - 0)

The IDEF0 diagram of the overall Integrated Construction Environment (ICE) is shown in Figure 5. It represents the highest possible abstraction of the ICOM which are required by and/or acting upon the ICE. The main input is the Construction Industry know-how (Design and Construction) and product's specifications. The former covers an abstraction of the construction industry know-how. The

comprehensibility of this abstraction depends on the scope and the objectives set for the ICE. The latter, on the other hand, refers to a project specific information for which the ICE will be applied to, i.e. project specific data which is normally obtained during the design stage of the project.

The execution of this process is constrained and controlled by building regulations, standards, procurement methods, etc. While the mechanism required to implement the system is represented by the Information Engineering Support Tools such as; CASE tools, object oriented environments, data definition languages, etc. The output of the system is a product specific data sets of which content depends on the main functionality of the ICE and its downstream applications.

The Decomposition of the Context Diagram (A0)

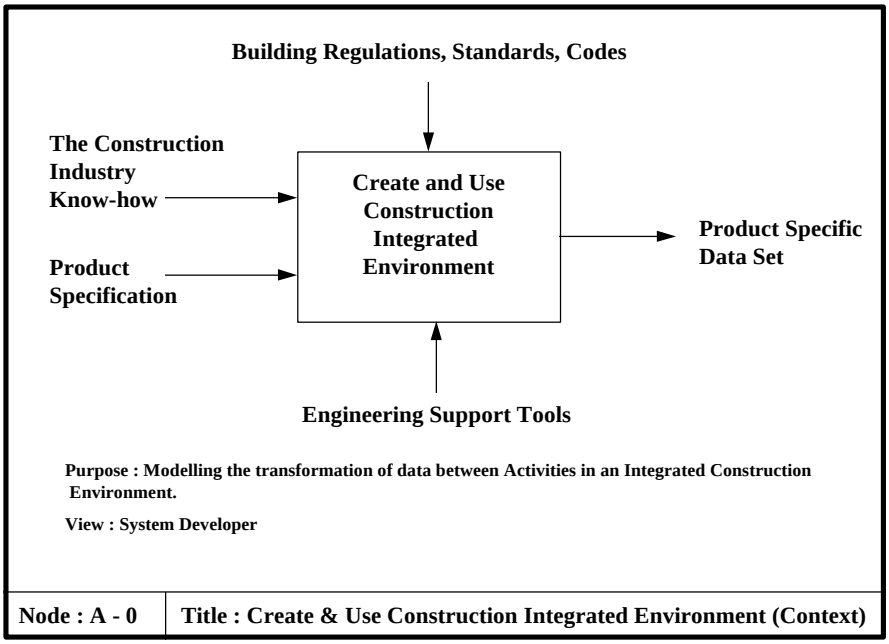

Figure 5: The Context Diagram of A-0

The initial decomposition of the Context Diagram is shown in Figure 6. The first activity is Define Product and Other Data Models. This activity requires developers to identify the individual parts (models) which are needed to satisfy the scope and the objectives of the ICE's main functionality. The output of such an activity is a set of data models representing a product data model along with other data models required by the ICE such as construction planning model, estimating model, material's specification model, etc. The performance of this activity, i.e. the behaviour of the various models, is subjected to constraints such as regulations, standards, procurement methods, etc. Moreover, specific constraints are also applied as and when required by

the requirements of a specific project. This is shown by the Feedback arrow which is the output of activity 3, Applied Project Specific Knowledge. Activity 1, can be implemented using any information engineering support tools.

The second activity is the Build and Integrate Applications. This activity develops downstream applications and integrates them within the ICE. Downstream applications refer to those developed externally using commercial software packages and/or internally using the ICE information engineering support tools. The integration process requires a well defined structure for the central core data models so as information required by various applications can be easily accessed while altered or updated information, by the applications, can be easily written back to the product model. For example, to obtain information about a number of design elements (instances) from the product model, an application needs to know where this information is stored and the type of data structure, e.g. list, array, record, etc. Hence, the data models become part of the mechanism for this activity i.e. applications utilise the data models to access the relevant information. The main input for this activity is the product specific data sets which are generated by activity 3, Apply Project Specific Information. For example, in order to produce a cost estimate for a project, the estimating application requires specific information about that project which must be presented in the required format. The output of this activity reflects the output of the applications which in turns can be used as a feedback to improve their future performance. Finally, the full implementation of such applications is subject to the relevant regulations and standards.

The final activity is Apply Project Specific Information. This activity populates and uses the previously developed data models with project specific information, i.e. instantiating instances normally from the design information. This implies that this activity uses data models as a mechanism to fulfil its objectives. It generates product specific data sets, as an output, and feeds them into activity 2 so that applications can be implemented. Therefore, once the project is defined, users can run the system to obtain the required output e.g. construction plans, estimates, Virtual Reality models, etc. Moreover, the output of this activity, i.e. project specific requirements, can be applied as constraints to the utilisation of the data models i.e. specific projects required specific data behaviour. The implementation of this activity, however, can only be carried out within an integrated environment, ICE.

In order to show the details of the above activities, each activity has been decomposed into its constituents sub-activities and are represented in low level IDEF0 Diagrams.

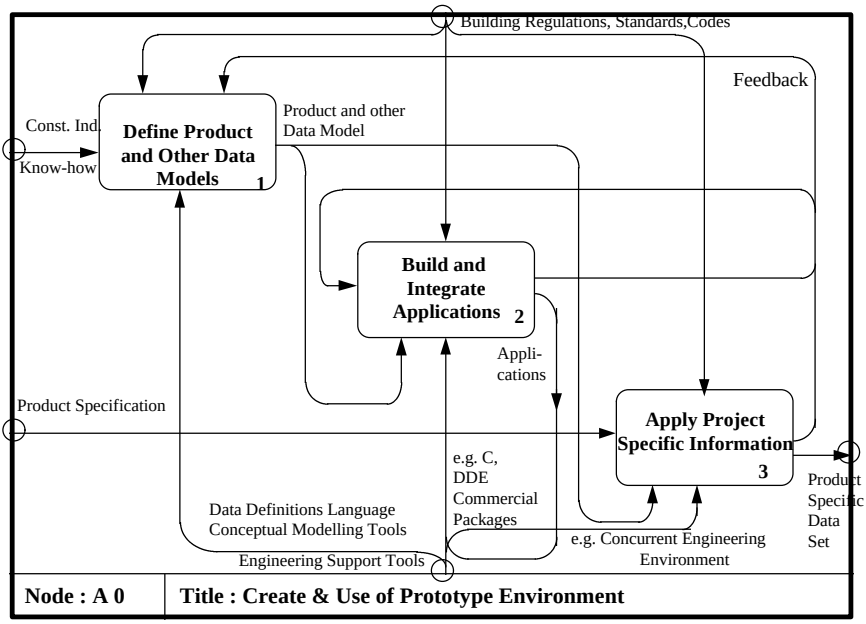

Figure 6: The Decomposition of the context diagram (A-0) - A0

Define Product and Other Data Models (A1)

Figure 7 shows the decomposition of Define Product and Other Data Models activity. Activity 1, in this diagram, is Define Product Data Model. It creates a product data model, based on the construction know-how, which represents various levels of data abstractions of the concerned building type, i.e. concrete buildings. Its main output is a defined and implemented product data model which satisfy the scope and the objectives of the ICE. It's implementation is materialised through information engineering support tools. Finally, its behaviour is subjected to constraints imposed by the various regulations and standards.

Activity 2 is Define Other Data Models. This activity produces data models which are related to one or more building types and fulfil the requirement of specific applications. The existence of such models do not depend on the existence of the product data model. For example, a Construction Planning Data Model which contains information about construction methods, contractor's resources, planning knowledge, etc. can exist before a product data model is defined. It can also be used to plan other type of buildings which are outside the scope of its ICE. These models use the construction know-how as their main input and produces specialised data models, e.g. Cost Estimating Data Model, Site Layout Data Model, Construction Planning Data Model, Specification Data Model, etc. Such models are also constrained by regulations and standards, and are implemented using information engineering support tools.

The data models are simply static representation of data. However, the interactions between the various data models are controlled and carried out by the applications

themselves. For example, a cost estimating application can interrogate the product model for instances of a building which in turns can activate certain methods to generate their associated specifications from a specification model.

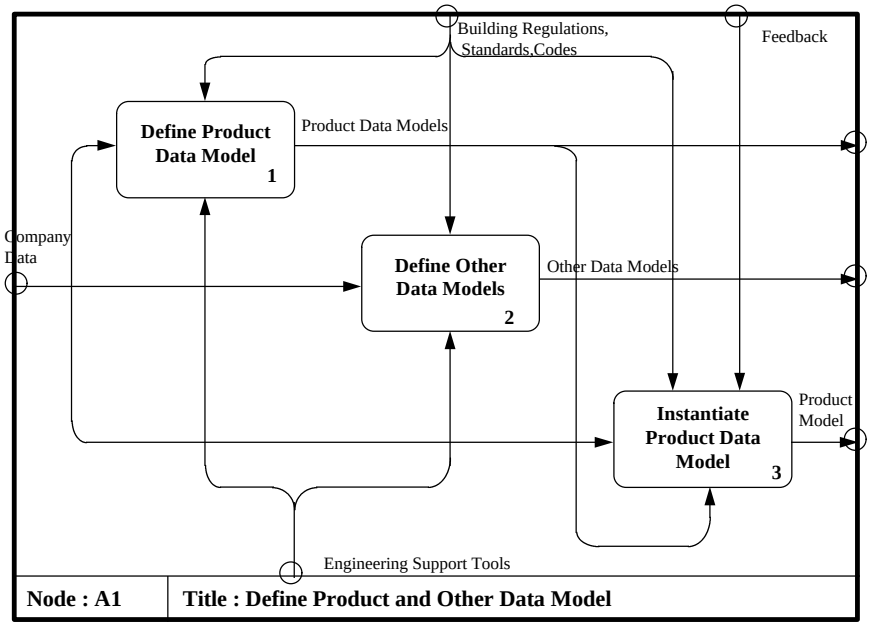

Figure 7: Define Product and Other Data Model - A1

The last activity in this diagram is activity 3, Instantiate Product Data Model. This activity defines the status of objects by filling in the range of values required by their attributes. This information can be accessed either from on-line or in-house databases. When the data is entered into the product data model, i.e. utilising product and other data models as a mechanism, it becomes a product model, i.e. producing product model as an output. The performance of this activity is subjected to regulations and standards as well as the feedback coming from applying specific knowledge into the product model i.e. activity 3 A0.

Build and Integrate Applications (A2)

This activity has been decomposed into three sub-activities as shown in Figure 8. Activity 1 is Build Applications. This activity refers to the process of developing any application that needs to be integrated with the ICE, i.e. with the central core data models. Applications can either be existing ones or specially developed for such integration. In either case, they are implemented using programming languages, commercial software packages, DDE, etc. The development of such applications are subjected to further feedback arising from the use of the ICE which can be used as an input to modify/extend the applications. Finally any function performed by an application should comply with regulations and standards.

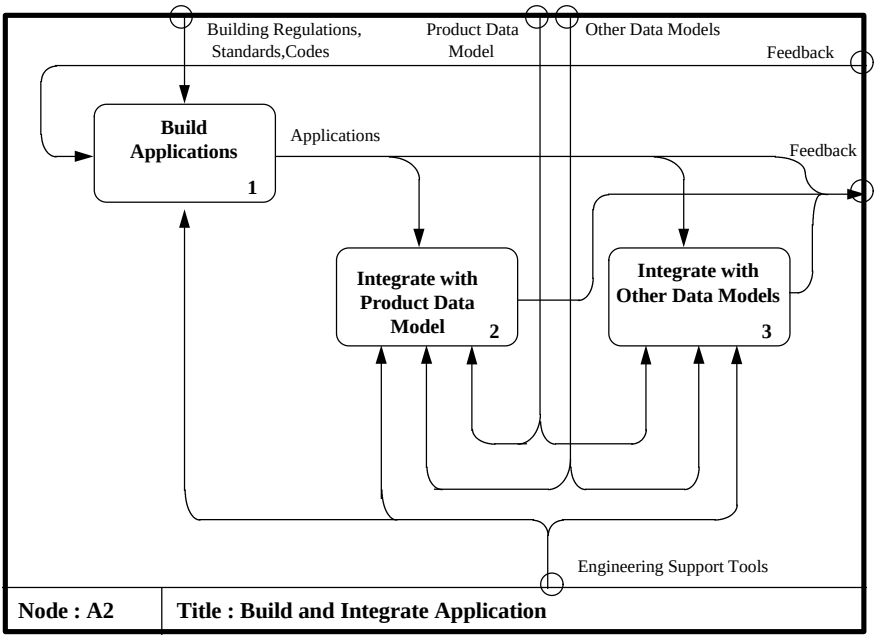

Figure 8: Build and Integrate Applications - A2

Activities 2 and 3 integrate the developed applications with the product data model and other data models, respectively. They are carried out through the development of interfaces between the applications and the environment with which ICE is implemented. Thus, the data models and programming facilities act as a mechanism to perform these activities. Moreover, the applications themselves create constraints on these two activities. This is because the data defined by the data models must satisfy all the requirements of these applications. The outcome of these activities are used as future feedback mechanism for activity 3 A0, Apply Project Specific Information. Also, it is used as a feedback which can either impose new constraints onto the data models, i.e. activity 1 A0, or be used for future developments of each application.

Apply Project Specific Information (A3)

This activity is shown in Figure 9 where it has been decomposed into three activities. Activity 1 is Define Project Data. The activity's main function is to populate the product model with project specific information i.e. transforming product specifications into Product Specific Data Sets. The latter sets are then fed into the various applications as and when required. Normally, this activity takes its input from a design application and in the format of building elements and their associated geometric information.

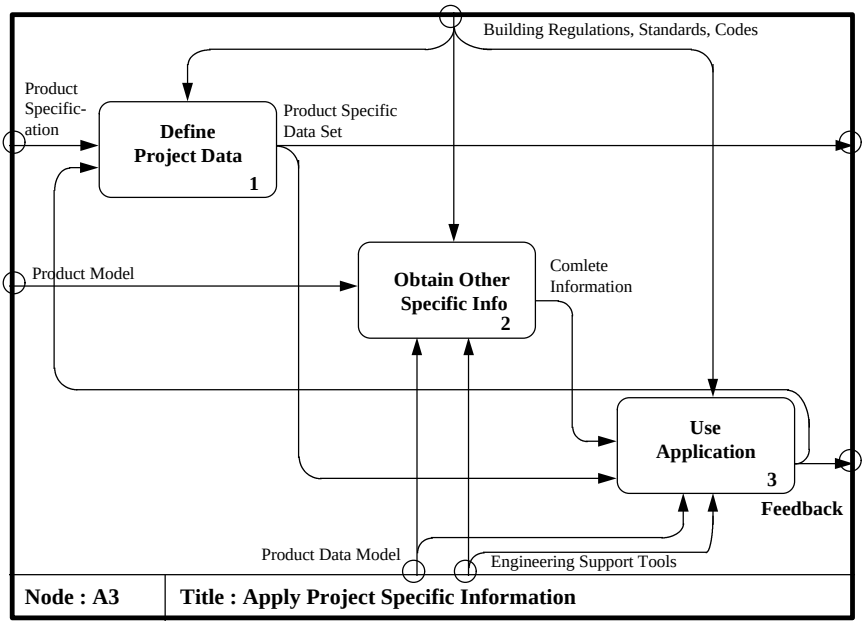

Figure 9: Apply project specific information - A3

Activity 2 is Define Other Specific Information. This activity obtains any other information which is required by the downstream applications from the other data models, i.e. where an application's data model does not fully provide the information required by the application. Therefore, the product model is used as the activity's main input. For example, for an application to determine whether a plant is required for a certain construction activity, it needs to know its size, location, site constraints, etc. before it can produce a recommendation. Such information is usually generated by interrogating the product model in the central core. This activity is normally application oriented one where its output satisfies the applications' requirements. It is again controlled by regulations and standards and implemented using the information engineering support tools.

The last activity is activity 3, Use Application. It takes the previously generated complete information sets as well as data from the populated data models as an input in order to allow the applications to perform their main functions. This activity is controlled by regulation and standards and implemented through either the information engineering tools or the applications' own environment. Its output is used as a feedback to enable users to change the Product Specification to improve the generated result.

IMPLEMENTATION

The above concept was implemented by the AIC research group (Automation and Integration in Construction) at the University of Salford. An integrated environment

"SPACE" (Simultaneous Prototyping for An integrated Construction Environment) has been developed running under Windows. This section addresses the implementation's view of SPACE.

At its current stage of development, SPACE integrates four external and one internal applications with central data models. The external applications are; design, construction planning, virtual reality modelling, and site layout planning, while the internal application is cost estimating. These applications have been implemented using commercial software i.e. Auto CAD for the design and site layout planning, Super Project Expert for the construction planning application, and World Tool Kit for the virtual reality application. KAPPA has Ben used as the information engineering support tool where all the data models are implemented. Moreover, the Cost estimating application has used KAPPA as its implementation media. The central core consists of:

1. A product data model which represents the feature and behaviour of concrete office buildings. It controls the flow of information between Auto CAD (as the design tool) and the central core (Alshawi and Pudra, 1994). This model acts as the main distributor of project's specific information to the other data models. The cost estimating application and the virtual reality application also use this model as their main "data model" to satisfy their main requirements.
2. A construction data model which represents generic construction activities, construction methods, contractors' available resources, etc. This model controls the communication between the central core and the Super Project Expert package. It generates the required construction activities along with their duration, resources, etc. and transfer them to the planning package (Alshawi and Hassan, 1994).
3. A site planning model which represents the know-how required to allocate temporary facilities on site as and when required by the construction site. It controls the information flow between the central core and AutoCAD (Alshawi and Suliman, 1994).

These data models are static representation of data i.e. they do not interact with each other. Interactions between the various data models are carried out and controlled by the applications. This means that if an application needs specific information from another data model than that of its own, it activates a function within its own data model which sends a message to the relevant data model asking for the required information. This approach can significantly control the development of the ICE and create an excellent maintenance strategy for the whole environment.

Figure 10, explains the flow of information within SPACE. In this Figure, the central core is presented by its components while the various applications are shown on the right hand side box. It can be clearly seen that applications require information from different data models. This is usually arise in an attempt to either eliminate data redundancy, i.e. adopting data sharing approach, or overcome dependency problems between the various applications. Both cases are important and have to be considered during the design and implementation of the ICE

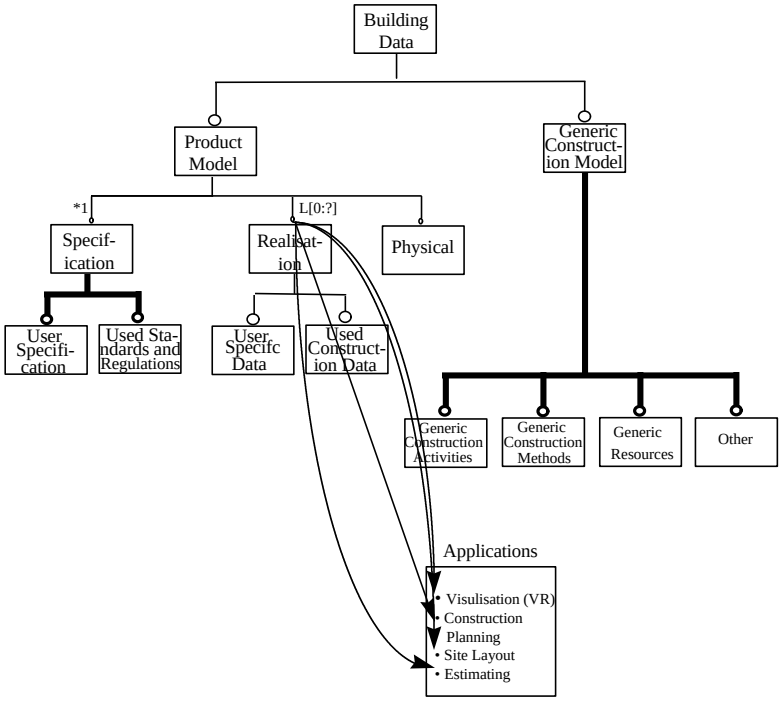

Figure 10: Linking of applications in SPACE

CONCLUSION

The complexity of construction applications and the vast amount of information involved in a project have hindered the development of an integrated environment for construction. The lack of a high level structure (a strategic vision) and a full understanding of such environment has led to the development of a number of small integrated applications in various fields of construction. This paper have addressed this issue and proposed a strategic, but generic, framework for an Integrated Construction Environment (ICE). The framework is based on three main components; the central core data models, project specific information, and the construction applications.

The paper has adopted IDEF0 notations to represent a process based model which shows the various activities involved in the ICE along with their relationships. The framework highlights a methodology for operating such environment and addresses three main issues; the data models, the applications, and the project specific information/requirements. Data models have to be developed first using an information engineering supporting tools. External or internal applications can then be either developed/linked with the central data models. Such an environment is triggered off by project specific information.

This framework has already been implemented by the AIC research group at the University of Salford with great success. Several applications, external and internal,

have been developed and integrated with the central data models. Applications have been developed individually and independently from each other. Attention were given to identify common functions (i.e. functions required by more than one application) in order to eliminate the duplication of these functions. The approach has proven to be essential to such development as it gives an excellent view on how the various parts of an ICE are integrated. Also, it could significantly improve the understanding of the working environment which could lead to meeting future users' needs and requirements.

REFERENCES

Alshawi, M., 1994, A Run Time Exchange Of Elemental Information Between Cad And Object Models: A Standard Interface, The International Journal of Construction Information Technology, 2 (2), pp 37-52.

McKay, A and Bloor M. S., 1991, The Role of Product Model in Effective CADCAM, Effective CADCAM 91: Taking Advantage of Engineering Technologies, pp 113-119.

Marca, D. A., and McGawn,C .L ., 1988, SADT: Structured Analysis and Design Technique, McGraw-Hill Book Company, New York.

Ross, D. T., 1977, Structured Analysis: A language for Communicating Ideas, IEEE Transactions on Software Engineering, SE-3(1), pp 16-34, January.

Ross, D. T., and Schoman Jr., J. K., 1977, Structured Analysis for Requirements Definition, IEEE Transaction on Software Engineering, SE-3(1), pp 16-34, January.

Dym, C. and Levitt, R., 1991, Knowledge-based Systems in Engineering, McGraw-Hill Inc., pp 182-183.

Alshawi M, and Hassan Z, "A Run Time Generation Of Construction Plans: Integrating Construction Information Model with Building Model", The First European Conference on Product and Process Modelling in the Building Industry, Dresden, Germany, October, 1994 (proceedings in print).

Alshawi M, and Pudra F, "Conceptual Modelling For 3d Simulation Of Design And Construction", Third Conference on Computer Applications in Civil Engineering, Kuala Lumpur, August, 1994 (Ref 4. proceedings in print).

Alshawi, M, and Suliman J, "Applying Structured Processes Analysis to Site Layout Planning" submitted for publication at the 6th International Conference on Computing in Civil and Building Engineering, Berlin, July 12-15, 1995.