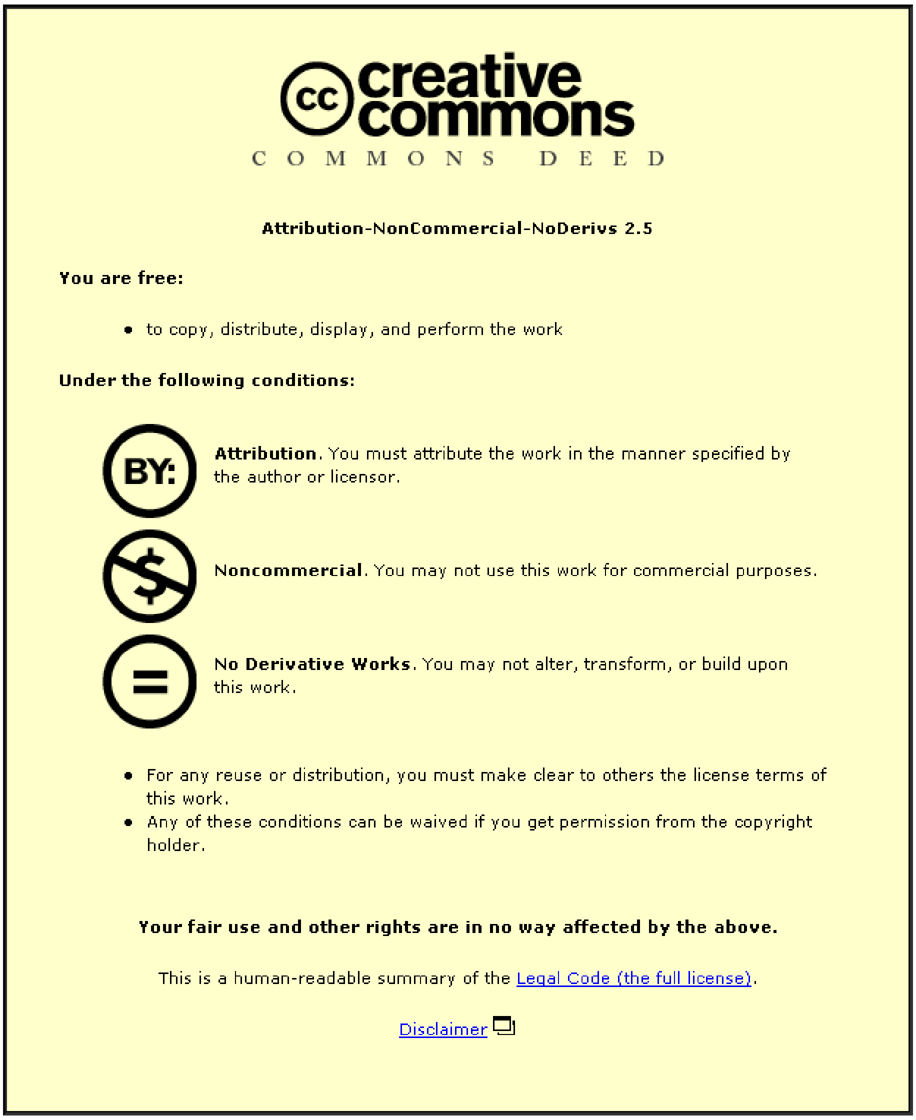


# Analytical Design Planning Technique (ADePT): A Dependency Structure Matrix Tool to Schedule the Building Design Process

**Simon Austin, Andrew Baldwin, Baizhan Li and Paul Waskett**
**Department of Civil and Building Engineering**
**Loughborough University**
**LE11 3TU, UK**

## Abstract

Current planning practice takes little account of the interdisciplinary, iterative nature of the building design process. This leads to a compromised design process containing inevitable cycles of rework together with associated time and cost penalties in both design and construction. The Analytical Design Planning Technique (ADePT) is a planning methodology which helps to overcome these difficulties. The central part of ADePT is a Dependency Structure Matrix (DSM). This paper describes DSM techniques and a tool developed to optimise the design process.

## Keywords

ADePT, design, planning, design management, matrix analysis

## Introduction

In recent times there has been a growing understanding of the importance of effective design management to facilitate a co-ordinated design within budget, and to ensure the smooth running of the project. Traditionally, building design has been planned by the same methods used to programme construction. These techniques do not allow the effect of variations and delays to be fully understood within an iterative process such as design. They monitor progress based upon the completion of drawing work and other design deliverables as opposed to the availability of key pieces of information. The ADePT methodology shown in Fig. 1 was devised by Newton (1995) to overcome these limitations, and associated computer tools have now been developed to facilitate more effective planning and management of building design. The first stage of the methodology is a model of the building design process, representing design activities and their information flows (inputs and outputs). These activities are responsible for producing, directly or indirectly, the design deliverables (calculations, drawings and specifications) and the information takes the form of design data. The data in this model is linked via a table showing the activities' dependency on information, to a Dependency Structure Matrix (DSM) analysis tool which is used in the second stage to identify iteration within the design process and schedule the activities with the objective of optimising the task order. The third stage of the methodology produces design programmes based on the optimised process sequence. The technique requires some iteration between the DSM and programming stages.

This paper describes a review of existing DSM techniques and tools, the establishment of a DSM tool to suit the building design process, the production of matrices of the building design process and the validation of these matrices via the testing of ADePT on historical and current building projects. Detailed descriptions of the process modelling and programming stages are given elsewhere (Austin *et al.*, 1998a, 1998b).

## DSM methodology

### Fundamentals

The success of network analysis and bar chart techniques to plan construction work has led to their extensive use in planning design. Such techniques are not suitable to plan design work because they schedule a process on the basis of the completion of elements of work rather than the production of information, and having been devised to plan sequential processes, they do not have the capability to deal with the iteration in a design process. In the 1960s, Steward developed a theory that a complex problem such as design could be more efficiently solved by representing the interrelationships between activities in the form of a matrix (Steward, 1965) which could then be used to decompose the problem, thus establishing the contributing sub-problems. Steward termed the technique Design Structure Matrix analysis, however recent work has demonstrated the application of the technique to problems outside of design, and hence it is now known as Dependency Structure Matrix analysis (Browning, 1997).

Fig. 2(a) shows a matrix for a very simple design problem. The problem contains 20 activities which are listed arbitrarily down the left hand side of the matrix. The same activity order is also listed across the top of the matrix. An assumption is made that the activities are undertaken in the order listed within the matrix, starting from the top left corner and finishing in the bottom right corner. Each mark in the matrix indicates that the activity on the left hand side is dependent upon the activity at the top of the matrix. This means that in the assumed order of activities, a mark below the diagonal shows that an activity is dependent on information which has been produced by a previous activity whereas a mark above the diagonal indicates that a activity is dependent on information that has yet to be produced. This can be overcome by estimating the information that is as yet unavailable and then verifying the estimate once the information generating activity has been undertaken. For example, in Fig. 2(a) it can be seen that activity E depends on some information from activity L that at the time has not been undertaken. If this information is estimated, activity E can be carried out and then activity L, following which the estimate can be verified. It may be that the activity dependent on the estimated information (activity E) has to be redone if the original estimate was not accurate, resulting in an iterative loop of design activities. In this case it involves at least eight activities (E to L), but possibly up to 15 as activity L in turn requires an estimate of information from activity S (hence the shaded block of tasks).

The need to estimate information and then carry out activities more than once results in any process being inefficient. It is desirable to reduce the need for estimates and therefore iteration within the process. This is achieved by reordering the activities within the matrix so that the marks are below the diagonal or as close to it as possible thus producing the optimal sequence of activities, a feature that is not guaranteed when using network analysis (Alkayyali *et al.*, 1993). The process of reordering the matrix is called partitioning. In this case (Fig. 2b) it can be seen that activity E is now in a smaller loop of only five tasks, significantly reducing the amount of potential repetition.

## Partitioning a Matrix

The purpose of partitioning a matrix is to maximise the availability of information required, and minimise the amount of iteration and the size of any iterative loops within the process. The process is ordered to minimise the number of dependencies above the diagonal. Fig. 2(b) shows the partitioned version of Fig. 2(a). It can be seen that the sequence is altered and that fourteen activities contribute to four iterative loops and therefore, in the process represented, some information estimates must be made.

Partitioning can be achieved through the employment of a technique called path searching (Steward, 1981) or through the application of a mathematical system such as Boolean algebra (Ledet and Himmelblau, 1970), a knowledge-based expert system (Rogers, 1989) or a genetic algorithm (McCulley and Bloebaum 1994, Rogers *et al.*, 1996).

Partitioning a matrix sequences activities that do not contribute to iterative loops and indicates the activities that are within iterative loops, but does not sequence the activities within the loops. This is because the activities that contribute to a loop are all interrelated and any of them can be the first activity undertaken in the completion of the loop. It is desirable that the activities within a loop be ordered to reduce the number of estimates that must be made. This forms the first part of a further process termed tearing.

## Tearing an Iterative Loop within a Matrix

Tearing a loop means reducing the size of the iteration by minimising feedbacks and identifying estimates that can be made with some confidence and that therefore do not need to be revisited as part of the iterative process. The first stage in the tearing process is the scheduling of activities within the loop to reduce the number of estimates that are required and to identify a starting point for the undertaking of the loop. This is done by either the preparation and analysis of a shunt diagram of the activities in the loop (Steward, 1981) or the application of an algorithm which adopts a heuristic approach to minimising the feedbacks in the loop (Rogers, 1989). This is demonstrated in the example where the last loop has been reordered to reduce the number of estimates from three (Fig. 2b) to two (Fig. 3). The second stage in tearing a loop is the removal of one or more information dependencies in order to reduce the size of the loop. Rogers' knowledge-based heuristic approach to tearing results in the removal of dependencies on the basis of an algorithmic calculation which determines which dependencies are most responsible for causing the loop. The shunt diagram method results in a series of suggested tears and a weighting of their effectiveness in reducing the size of the iteration, however a decision is required by the user before a tear is made. This second approach means that a knowledge of the problem is required by the user, however a more meaningful analysis of the problem is usually achieved because an assessment is made regarding the feasibility of each tear based on practical experience (Steward, 1993). An example of a matrix with ordered activities in a loop resulting from a shunt diagram is given in Fig. 3, based on the partitioned matrix in Fig. 2(b). All five tasks contribute to the longest circuit within the loop, called the principal circuit, so tearing the information dependencies in any of the rows or columns in the loop will have the effect of reducing the size of the iteration. The shunt diagram method suggests that tearing the link between tasks P and N will have the greatest effect.

### Manipulating a Matrix with Classified Information

There are many information dependencies between activities in complex problems such as building design. The resulting matrix can be clarified by accounting for different levels of information importance and therefore strengths of dependency. This is done by classifying the dependencies within the matrix and using a partitioning algorithm that can prioritise the sequencing of activities on the basis of these classification. The classification of information has the same effect as tearing a matrix but is carried out prior to the production of the matrix. Following the classification of information in a matrix, further tearing may be necessary in order that the highly complex design process is decomposed into manageable sub-problems.

The classification of information within a matrix is a highly subjective exercise. A number of scales aimed at the classification of information in a DSM have been devised. Rogers and Bloebaum (1994) advocate the development of a seven point scale of design information dependency strengths which can either be subjectively determined by a design manager or calculated by an algorithm. Smith and Eppinger (1993) give details of alternative scales, one with a percentage weighting and one with a three point scale of dependencies in iterative loops to indicate the probability of a dependency contributing to the iteration. Austin *et al.* (1996) describe a further three point scale of classifications based on the strength of dependency of information, sensitivity of activities to changes in information, and the ease with which information can be estimated.

## Application of DSM

### Use of DSM in Manufacturing and Engineering

Interest shown in DSM has been largely limited to academia. Although the theory of DSM has been applied in a number of circumstances, analyses are only just now being undertaken in practice. Much of the work to date has focused on the optimisation of design problems in engineering applications.

Rogers (1989) and other researchers at the NASA Langley Research Centre have demonstrated how DSM could be applied to design problems faced at NASA. These demonstrations have concentrated on the scheduling of problems with up to 50 activities at the conceptual phase of the design process. The focus of Rogers' research has been on the development of a number of extensions to Steward's basic DSM theories, and a software tool called DeMAID/GA incorporating these features. The main DSM feature that Rogers has developed is a facility to sequence the tasks in a design project with the goal of optimising the project duration or cost, characteristics that are considered desirable in many projects. To achieve this, a duration and cost must be allocated to each design task, an exercise that is undertaken in the planning of most engineering and construction projects. In most instances, a cost allocation will be based on the estimated time to complete the activity. The calculation of times and costs of groups of tasks that are not involved in iterative work is a simple summation, however determining the overall duration and cost of an iterative loop of tasks is a more complex operation.

At the Sloan School of Management, MIT, Eppinger and his group of co-researchers have also been responsible for developing extensions to Steward's work. They have applied these to design problems in a range of engineering environments. In addition to the work done in information classification systems, described above, Eppinger has introduced the concepts of 'parallelisation' and 'artificial decoupling' to simplify the design process (Eppinger *et al.,* 1990). Parallelisation involves an examination of a DSM and the identification of activities that can be performed concurrently. These activities are not related but the final task in the sequence that they depend on will be the same. The identification of these activities allows them to be programmed to be undertaken simultaneously. Artificial decoupling entails the removal of dependencies from the matrix to reduce the size of iterative loops in the same way as Steward's tearing. However, a dummy activity is added to the matrix and linked to the two activities that were previously interrelated, thus indicating that an information estimate is required.

The MIT group has applied DSM to a number of engineering problems, including semiconductor design (consisting of 60 activities) for Intel Corporation, and an automotive brake system design (consisting of over 100 activities) for General Motors. In a separate exercise, McCord and Eppinger (1993) analysed problems of automotive engine design for General Motors and laptop computer development, while Pimmler and Eppinger (1994) also used DSM to examine the design process of a climate control system for the Ford Motor Company.

### Use of DSM in the Construction Industry

To date, the application of DSM to problems in the construction industry has been limited to research work at Loughborough University and VTT in Finland.

In early work on the ADePT methodology, Austin *et al.* (1994) described the use of DSM in a simple building design problem comprising some 50 activities across the architectural, civil engineering and structural engineering disciplines. This work led to the conclusion that DSM is a tool that could be used to demonstrate areas in a design that need to be undertaken in an iterative manner. In the simple example considered, these areas of iterative work reflected the parts of the building that would typically require close co-ordination and redesign to be undertaken. This finding validated the use of DSM in the simple example considered. Also, Baldwin *et al.* (1995) described how DSM could be applied to simple problems in the scheme stage of a building's design. Discrete event simulation was then applied to predict the affects of changes in the design with regard to the overall duration and resource allocation of the process (Baldwin *et al.,* 1998).

At VTT in Finland, Huovila *et al.* (1995) demonstrated the application of DSM on a design problem in construction, comprising some 30 design activities. The process was analysed at an overview level, the feasibility of DSM to solve design problems in construction was tested, and it was concluded that the technique can be effectively used in construction to find better sequences of design tasks. Further applications of DSM have been demonstrated at VTT, such as the application of the technique to schedule work across all stages of a construction project (Vahala, 1997). Again, this work has been focused on problems at an overview level.

### Linking the process model to a DSM tool with information classifications

In the ADePT methodology, the information to be represented and analysed by a DSM is contained in the design process model that forms the first stage of the methodology (Austin *et al.,* 1998a). This means that the model database must be formatted in a manner that is compatible with the DSM tool, which has been achieved via a dependency table.

The second stage of this process is the allocation of classifications to the information flows in the model, so that the DSM tool can prioritise dependencies while partitioning the matrix. A three-point dependency classification system was devised specifically as an appropriate means of rating information within the building design process (Fig. 4).

This system of information classification relies on three factors (Fig. 5): the strength of information dependency; sensitivity to change of information; and the ease of estimating information. This means that three separate subjective judgements must be made by designers and resulting classification is given a rating of either 'A', 'B' or 'C' (where 'A' = strong .... 'C' = weak). The philosophy adopted by ADePT is that weak dependencies (classified C) can be omitted from the matrix partitioning (because an accurate estimate can easily be made), and therefore the size of iterative loops can be reduced and the design process clarified.

Following the compilation of a generic design process model, generic information classifications were determined and validated through interviews with practising designers. Where a generic classification could not be determined, the designers selected from a likely range, i.e. A/B, B/C and A-C. The information classifications were then added to the dependency table. As part of the process of planning a project with ADePT, the generic dependency table and classifications must be re-evaluated to make it project-specific. It is then formatted and imported to a DSM tool, thus representing it in matrix form.

## A prototype DSM tool

### A review of existing DSM programs

Two DSM tools have been reviewed through their practical application to design problems: the Problem Solving Matrix (PSM) and Design Manager's Aid for Intelligent Decomposition with a Genetic Algorithm (DeMAID/GA).

PSM was developed by Steward, based around his DSM theories. The tool operates in a Windows format and a matrix is set up through either the direct input of activities and dependencies via the program's user interface, or by importing the data in a matrix form in DSS, CSV or MDB formats. In its latest form (version 3.33), the program can manipulate large matrices: testing has shown that a design process in excess of 700 activities can be analysed, and information dependencies can be input with a ten-point range of classifications. The tool allows tasks and dependencies to be added or deleted at any time, can partition and suggest places to tear the matrix, and clearly indicates dependencies and iterative loops of tasks. Fig. 6 shows a matrix produced with PSM for a case study project described later. A further feature is the ability to trace affected tasks following a change in the matrix. Operating the tool and interpreting its output is easy, however it can take up to five minutes of calculation time when analysing large design problems.

The Design Manager's Aid for Intelligent Decomposition with a Genetic Algorithm (DeMAID/GA) is a tool developed at the Langley Research Centre at NASA (Rogers, 1996). It is a program that operates in Macintosh and UNIX environments, which can analyse matrices of 85 and 200 activities respectively, incorporating a seven-point information classification system. Data is input via the compilation of a text file describing features and relationships of each design task; however this is not a particularly simple process because it involves the transfer of data between files. The presentation and operation of the tool vary from Steward's DSM theory in a number of ways including partitioning the matrix to achieve as few dependencies as possible below the diagonal, rather than above it. Dependencies are not shown simply by a mark in the matrix, rather by the intersection between a horizontal line from one activity and a vertical line from another indicating an output from the first activity is required as an input to the second. Fig. 7 shows these fundamental differences in the way the matrices are represented.

DeMAID/GA offers a number of additional features, the main one being the scheduling of a design project with the goal of optimising the project duration or cost, as described above. Manipulating a matrix, like inputting data, involves the transfer of data between files which can become a laborious exercise; however the main drawback for our research is its inability to analyse large matrices, and hence schedule the detailed building design process model in the first stage of the ADePT methodology.

### Algorithmic Matrix Manipulation Program (AMMP)

The review of DSM techniques led to the production of a specification of a matrix analysis tool to suit the characteristics of the design process in a construction project. This specification is not presented here due to its commercial sensitivity, however it contained details of essential and desirable features of the tool. Following the review of existing programmes, it was decided that a DSM tool that could work effectively in the ADePT environment should be developed because although PSM and DeMAID/GA are effective at analysing certain design problems and contain a range of features, a tool that could analyse large-scale problems with enhanced graphical output, incorporating the essential features in the specification was necessary. Accordingly, the Algorithmic Matrix Manipulation Program (AMMP) has been developed, although to date some of the desirable features identified in the specification have not been incorporated.

The program employs the same matrix manipulation algorithm as PSM. It can be seen from Fig. 8 that the program's output is clear, indicating information dependencies with a coloured mark annotated with an A, B or C and highlighting iterative loops of tasks. Also, dependencies that are modified following the production of a matrix are highlighted differently, allowing them and their effects to be easily monitored.

The tool is capable of dealing with a design process comprising over a thousand activities. Data is inputted to the matrix via a spreadsheet containing details of activity relationships and information dependency classifications in the format of the ADePT dependency table. This means that information can be transferred from the design process model to AMMP very quickly and easily, although additional tasks and dependencies can be added directly through the program's user interface if necessary. Matrix partitioning takes a few seconds to complete following the selection of the operation from a pull-down menu.

Another feature of AMMP is the facility to view two matrices simultaneously so that the effects of changes to dependencies and their classifications can be easily reviewed. Also, a part of a matrix can be partitioned so that the effects of changes in the design process following the completion of some of the design can be simulated. This feature is shown schematically in Fig. 9.

Functionality is currently being added to AMMP. This includes a user interface that allows the DSM tool to be fully integrated with the process model, dependency table and programming stage of ADePT. This will allow the effects of changes to the design to be reviewed on either the matrix or programme, and changes imposed on the matrix to be more easily viewed on a design programme.

### The detailed building design process as represented on a DSM

Both AMMP and PSM have been used to analyse building design processes as part of the ADePT methodology. This work has involved the production and review of matrices for the design of a range of projects.

### Establishing a manageable design process

DSM analysis of large scale design problems has shown that, on the basis of the information classifications established prior to the analysis, the process contains a large loop of iterative work. In some cases this loop can consist of around 60% of the tasks in the process and must therefore be broken down to represent design problems of a more manageable size. This is achieved through tearing of the loop. This approach means that some information that had been deemed necessary (classified ‘A’ or ‘B’) must now be estimated (classified ‘C’). Such information cannot easily be estimated and so particular care must be taken in their selection. They must incorporate an appropriate margin for error so that these estimates do not need to be revisited during later stages of the design (i.e. we do not allow them to be responsible for the production of large iterative loops). This results in an additional cost to the building because it is over-designed. Analysis to date has shown that the building design process can be broken down to comprise between seven and twelve iterative loops, each typically consisting of between five and thirty interrelated tasks.

Establishing the effects of tearing an information dependency can prove difficult when a loop consists of a large number of tasks and interrelationships. The DSM can be viewed at a higher, less detailed level so that the interrelationships between elements of the building can be more easily understood. This is achieved by importing information from the dependency table to the DSM tool in a manner that groups the very detailed tasks in the model under the headings of systems and elements of the building. This results in a matrix of approximately 90 to 130 activities, each corresponding to a system, which is easier to interpret than the large scale matrix and, therefore, effective tears can be more easily determined. The insights that this analysis gives can help to identify information dependencies that can be torn in the matrix containing detailed task information.

### Interpreting the DSM output

A partitioned matrix indicates the optimal sequence of activities and their information dependencies. A more meaningful interpretation of the process can be found by representing information in the matrix on a programme which requires the duration of activities to be assigned and clearly indicates where tasks can be undertaken in parallel. This is achieved by transferring data into a project management package.

Analysis to date has shown that the iterative loops within matrices relate to design co-ordination issues, such as ceiling, underground services and perimeter structure co-ordination. The formatting of information in a matrix prior to its representation on a programme accounts for the iteration in the process and ensures that tasks in a loop are programmed to be undertaken concurrently so that co-ordination can be achieved (Austin *et al.,* 1998b).

### Validation and Application

The DSM analysis stage of ADePT has been verified by comparing the output from AMMP, PSM and DeMAID/GA against each other and against a matrix that has been partitioned by hand employing a Boolean algebra technique. All stages of ADePT have been validated through their application on a series of design projects, full details of which are published elsewhere (Austin *et al*., 1998a, 1998b). Within the DSM stage of the methodology, both AMMP and PSM have been tested on projects (Table 1) covering values from £16M to £35M. Figure 6 shows an example of a matrix from one of the projects examined. The output that has been generated, and the ease with which ADePT has been applied, indicate that the technique can be applied within an acceptable timeframe. Currently, a hospital project is being examined, involving a design process of around 800 tasks and 10,000 dependencies.

The first three projects had been recently designed, the output from the DSM tools and corresponding design programme to be compared with the planning that was undertaken in practice. This has shown that the latter did not take full account of the iteration within the design process, and that the design had been planned almost entirely to suit the construction process. Design related problems on site are currently being reviewed and work is being undertaken to determine whether these problems could have been avoided through effective planning with ADePT.

ADePT is being applied currently to the design of a £130M hospital to highlight potential areas of iterative work, determine co-ordination issues, and demonstrate the requirements for estimating information prior to the detailed design. This has involved active participation from the design team in developing a model and matrix. Results to date show that the time and effort required to develop a matrix is not excessive, and the output from the matrix is a useful means of understanding the design process across an entire project.

## Conclusions

This paper has described how, as part of ADePT, the output from a design process model is used to produce a dependency structure matrix (DSM) via a schedule of information requirements. A prototype DSM tool, the Algorithmic Matrix Manipulation Program (AMMP) has been developed that suits the requirements of building design analysis. The tool schedules tasks in the design process on the basis of their information requirements, identifies areas of iterative work within the process, and allows the effectiveness of eliminating the need for some information estimates by over-designing to be examined. The output is used to produce a design programme of the optimised design process.

The AMMP and PSM programs have been verified and tested by successfully representing the design process of a number of building projects within an acceptable timescale. The design activities and information dependencies in complex building projects can be represented on a DSM because the design process model in the first stage of ADePT covers a wide range of building systems. Practising designers and design managers shown the DSM methodology and tools have been enthusiastic about the effectiveness of the approach and the detailed level of information that is represented: a recently published summary of ADePT (Institution of Civil Engineers, 1998) has generated interest from design managers from consulting, client and contracting backgrounds, totalling some 70 companies, reflecting the industry's need for improved design planning. Work will continue to develop AMMP, and test the tool and overall ADePT methodology on further building projects.

## Acknowledgements

This work has been undertaken as part of a project entitled ‘Design Information Methodology and Tools for the Management of Detailed Building Design’. The research is funded under research grant GR/K74197 by the EPSRC, DETR and industry (AMEC Design and Management, Ove Arup and Partners, BAA, Boots, Laing Management and Sheppard Robson). The researchers wish to thank Don Steward and Jim Rogers for supplying PSM and DeMAID/GA for evaluation and testing.

## References

Alkayyali, O.J., Mansour, W.O., & Minkarah, I.A. (1993) How effective is CPM in planning and controlling construction projects?. *CIB W-65 Conference*, Trinidad W.I., September, pp 849-859.

Austin, S., Baldwin, A. & Newton, A. (1994) Manipulating the flow of design information to improve the programming of building design. *Construction Management and Economics*, **12** (5), pp 445-455.

Austin, S., Baldwin, A. & Newton, A. (1996) A data flow model to plan and manage the building design process. *Journal of Engineering Design*, **7** (1), pp 3-25.

Austin, S., Baldwin, A., Li, B. & Waskett, P. (1998a) Analytical Design Planning Technique (ADePT): an IDEF0v model of the detailed building design process. Accepted for publication in *Design Studies*.

Austin, S., Baldwin, A., Li, B. & Waskett, P. (1998b) Analytical Design Planning Technique (ADePT): programming the building design process. Accepted for publication in *Structures & Building*.

Baldwin, A., Austin, S., Thorpe, A. & Hassan, T. (1995) Simulating quality within the design process, *American Society of Civil Engineers (ASCE) Second Congress on Computing in Civil Engineering*, Atlanta, Georgia, June, pp 1475-1482.

Baldwin, A., Austin, S., Hassan, T. & Thorpe, A. (1998) Planning building design by simulating information flow. Accepted for publication in *Journal of Automation in Construction*, October, 1998.

Browning, T.R. (1997) Dependency structure matrix-based systems engineering, organisation planning, and schedule management. *MIT Lean Aircraft Initiative Working Paper*.

Eppinger, S., Whitney, D., Smith, R., & Gebala, D. (1990) Organising the tasks in complex design projects. *ASME Conf. on Design Theory and Methodology,* Sept., New York.

Eppinger, S.D., Whitney, D.E., Smith, R.P. & Gebala, D.A. (1994) A model-based method for organising tasks in product development. *Research in Engineering Design*. **6** (1), pp 1-13.

Huovila, P., Koskela, L., Lautanala, M. & Tanhuanpaa, V-P. (1995) Use of the design structure matrix in construction. *Proc. of 3rd International Workshop on Lean Construction*, Albuquerque.

Institution of Civil Engineers (1998) An innovative approach to design management. *Research Focus*, **33**, May 1998, p4.

Ledet, W.P. & Himmelblau, D.M. (1970) Decomposition procedures for the solving of large scale systems. *Advances in Chemical Engineering*, **8**, pp 185-254.

McCulley, C & Bloebaum, C.L. (1994) *Optimal Sequencing for Complex Engineering Systems Using Genetic Algorithms*. AIAA Paper No. 94-4327.

Pimmler & Eppinger (1994) Integration analysis of product decompositions, Proc. of *Sixth International Conference on Design Theory and Methodology*, Minneapolis, MN, Sept. 1994.

Rogers, J.L. (1989) *DeMAID - A Design Manager's Aid for Intelligent Decomposition User's Guide*, Technical Memorandum 101575, NASA.

Rogers, J.L. & Bloebaum, C.L. (1994) *Ordering Design Tasks Based on Coupling Strengths*. AIAA Paper No. 94-4326.

Rogers, J.L. (1996) *DeMAID/GA User's Guide - Design Manager's Aid for Intelligent Decomposition with a Genetic Algorithm*, Technical Memorandum 110241, NASA.

Rogers, J.L., McCulley, C.M. & Bloebaum, C.L. (1996) *Integrating a Genetic Algorithm into a Knowledge-Based System for Ordering Complex Design Processes*. Technical Memorandum 110247, NASA.

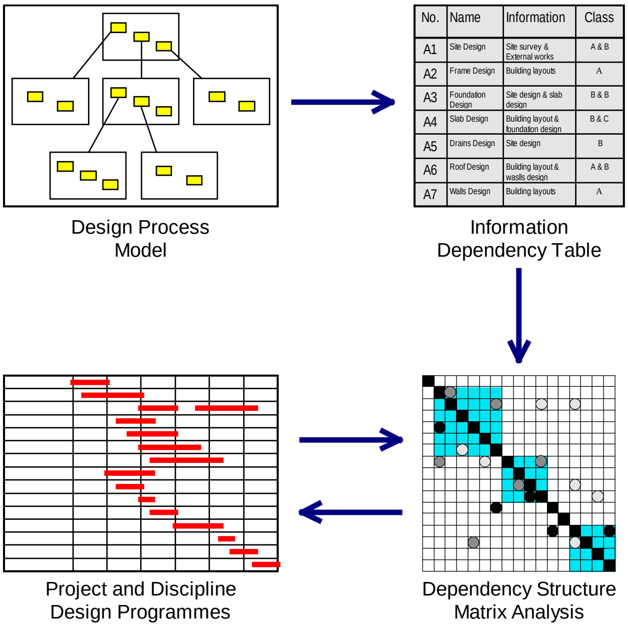
Smith, R.P, & Eppinger, S.D. (1993) Characteristics and models of iteration in engineering design. *International Conference on Engineering Design (ICED '93)*, The Hague, August 1993, pp 564-571.

Steward, D.V. (1965) Partitioning and tearing systems of equations. *SIAM journal on Numerical Analysis*, **2** (2), pp 345-365.

Steward, D. (1981) *Analysis and Management: Structure, Strategy and Design*, Petrocelli Books, USA.

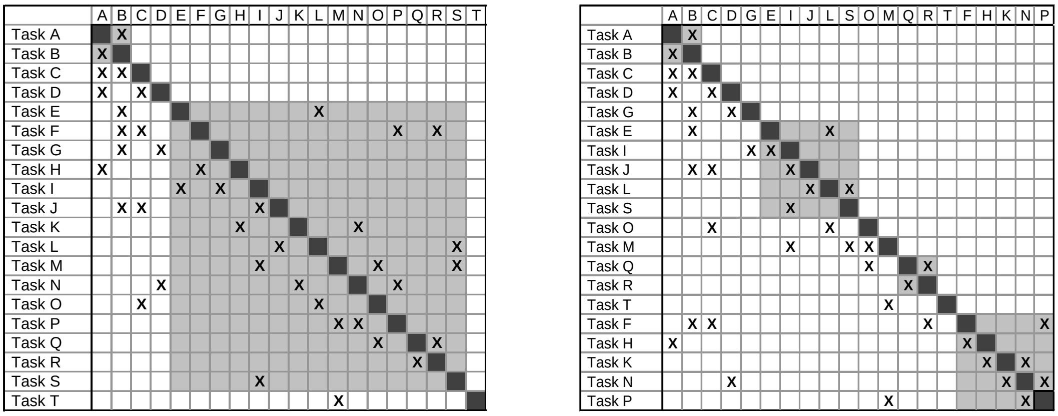
Steward, D.V. (1993) Re-engineering the design process. *Proceedings of the Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Morgantown, WV, April, 1993.

Vahala, P. (1997) *Prototype Tool for Construction Process Modelling and Management*. http://www.vtt.fi/cic/projects/star/star1/103.html



| No. | Name | Information | Class |
|---|---|---|---|
| A1 | Site Design | Site survey & External works | A & B |
| A2 | Frame Design | Building layouts | A |
| A3 | Foundation Design | Site design & slab design | B & B |
| A4 | Slab Design | Building layout & foundation design | B & C |
| A5 | Drains Design | Site design | B |
| A6 | Roof Design | Building layout & waslls design | A & B |
| A7 | Walls Design | Building layouts | A |

Fig. 1 Analytical Design Planning Technique (ADePT)



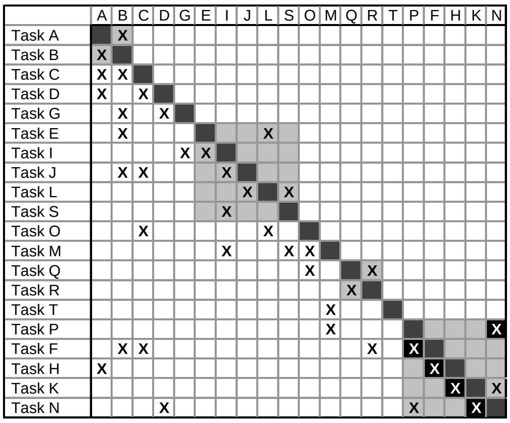
(a)

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Task A | ■ | X | | | | | | | | | | | | | | | | | | |
| Task B | X | ■ | | | | | | | | | | | | | | | | | | |
| Task C | X | X | ■ | | | | | | | | | | | | | | | | | |
| Task D | X | | X | ■ | | | | | | | | | | | | | | | | |
| Task E | | X | | | ■ | | | | | | | X | | | | | | | | |
| Task F | | X | X | | | ■ | | | | | | | | | | X | | X | | |
| Task G | | X | | X | | | ■ | | | | | | | | | | | | | |
| Task H | X | | | | | X | | ■ | | | | | | | | | | | | |
| Task I | | | | | X | | X | | ■ | | | | | | | | | | | |
| Task J | | X | X | | | | | | X | ■ | | | | | | | | | | |
| Task K | | | | | | | | X | | | ■ | | | X | | | | | | |
| Task L | | | | | | | | | | X | | ■ | | | | | | | X | |
| Task M | | | | | | | | | X | | | | ■ | | X | | | | X | |
| Task N | | | | X | | | | | | | X | | | ■ | | X | | | | |
| Task O | | | X | | | | | | | | | X | | | ■ | | | | | |
| Task P | | | | | | | | | | | | | X | X | | ■ | | | | |
| Task Q | | | | | | | | | | | | | | | X | | ■ | X | | |
| Task R | | | | | | | | | | | | | | | | | X | ■ | | |
| Task S | | | | | | | | | X | | | | | | | | | | ■ | |
| Task T | | | | | | | | | | | | | X | | | | | | | ■ |

(b)

| | A | B | C | D | G | E | I | J | L | S | O | M | Q | R | T | F | H | K | N | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Task A | ■ | X | | | | | | | | | | | | | | | | | | |
| Task B | X | ■ | | | | | | | | | | | | | | | | | | |
| Task C | X | X | ■ | | | | | | | | | | | | | | | | | |
| Task D | X | | X | ■ | | | | | | | | | | | | | | | | |
| Task G | | X | | X | ■ | | | | | | | | | | | | | | | |
| Task E | | X | | | | ■ | | | X | | | | | | | | | | | |
| Task I | | | | | X | X | ■ | | | | | | | | | | | | | |
| Task J | | X | X | | | | X | ■ | | | | | | | | | | | | |
| Task L | | | | | | | | X | ■ | X | | | | | | | | | | |
| Task S | | | | | | | X | | | ■ | | | | | | | | | | |
| Task O | | | X | | | | | | X | | ■ | | | | | | | | | |
| Task M | | | | | | | X | | | X | X | ■ | | | | | | | | |
| Task Q | | | | | | | | | | | X | | ■ | X | | | | | | |
| Task R | | | | | | | | | | | | | X | ■ | | | | | | |
| Task T | | | | | | | | | | | | X | | | ■ | | | | | |
| Task F | | X | X | | | | | | | | | | | X | | ■ | | | | X |
| Task H | X | | | | | | | | | | | | | | | X | ■ | | | |
| Task K | | | | | | | | | | | | | | | | | X | ■ | X | |
| Task N | | | | X | | | | | | | | | | | | | | X | ■ | X |
| Task P | | | | | | | | | | | | X | | | | | | | X | ■ |

Fig. 2 A simple example of a Dependency Structure Matrix



| | A | B | C | D | G | E | I | J | L | S | O | M | Q | R | T | P | F | H | K | N |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Task A | ■ | X | | | | | | | | | | | | | | | | | | |
| Task B | X | ■ | | | | | | | | | | | | | | | | | | |
| Task C | X | X | ■ | | | | | | | | | | | | | | | | | |
| Task D | X | | X | ■ | | | | | | | | | | | | | | | | |
| Task G | | X | | X | ■ | | | | | | | | | | | | | | | |
| Task E | | X | | | | ■ | | | X | | | | | | | | | | | |
| Task I | | | | | X | X | ■ | | | | | | | | | | | | | |
| Task J | | X | X | | | | X | ■ | | | | | | | | | | | | |
| Task L | | | | | | | | X | ■ | X | | | | | | | | | | |
| Task S | | | | | | | X | | | ■ | | | | | | | | | | |
| Task O | | | X | | | | | | X | | ■ | | | | | | | | | |
| Task M | | | | | | | X | | | X | X | ■ | | | | | | | | |
| Task Q | | | | | | | | | | | X | | ■ | X | | | | | | |
| Task R | | | | | | | | | | | | | X | ■ | | | | | | |
| Task T | | | | | | | | | | | | X | | | ■ | | | | | |
| Task P | | | | | | | | | | | | X | | | | ■ | | | | X |
| Task F | | X | X | | | | | | | | | | | X | | X | ■ | | | |
| Task H | X | | | | | | | | | | | | | | | | X | ■ | | |
| Task K | | | | | | | | | | | | | | | | | | X | ■ | X |
| Task N | | | | X | | | | | | | | | | | | X | | | X | ■ |

Fig. 3 The example matrix with suggested tear in last block

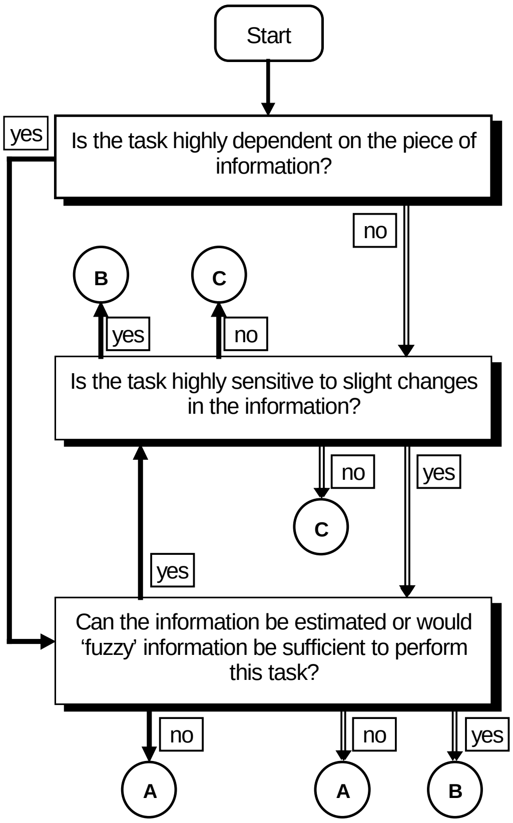
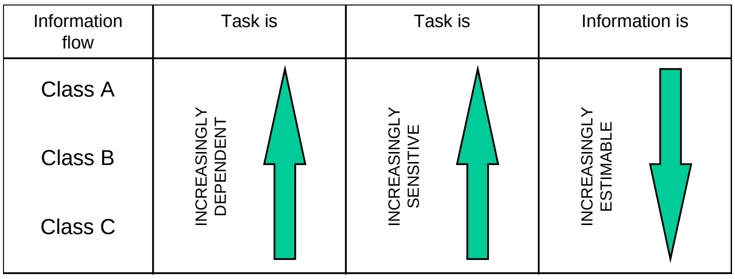

Start
Is the task highly dependent on the piece of information?
yes
no
B
C
yes
no
Is the task highly sensitive to slight changes in the information?
no
yes
C
yes
Can the information be estimated or would ‘fuzzy’ information be sufficient to perform this task?
no
no
yes
A
A
B

Fig. 4. Decision system for allocating information classifications (Austin *et al.* 1996)



| Information flow | Task is | Task is | Information is |
|---|---|---|---|
| Class A<br>Class B<br>Class C | INCREASINGLY DEPENDENT | INCREASINGLY SENSITIVE | INCREASINGLY ESTIMABLE |

Fig. 5 Characteristics of each information classification

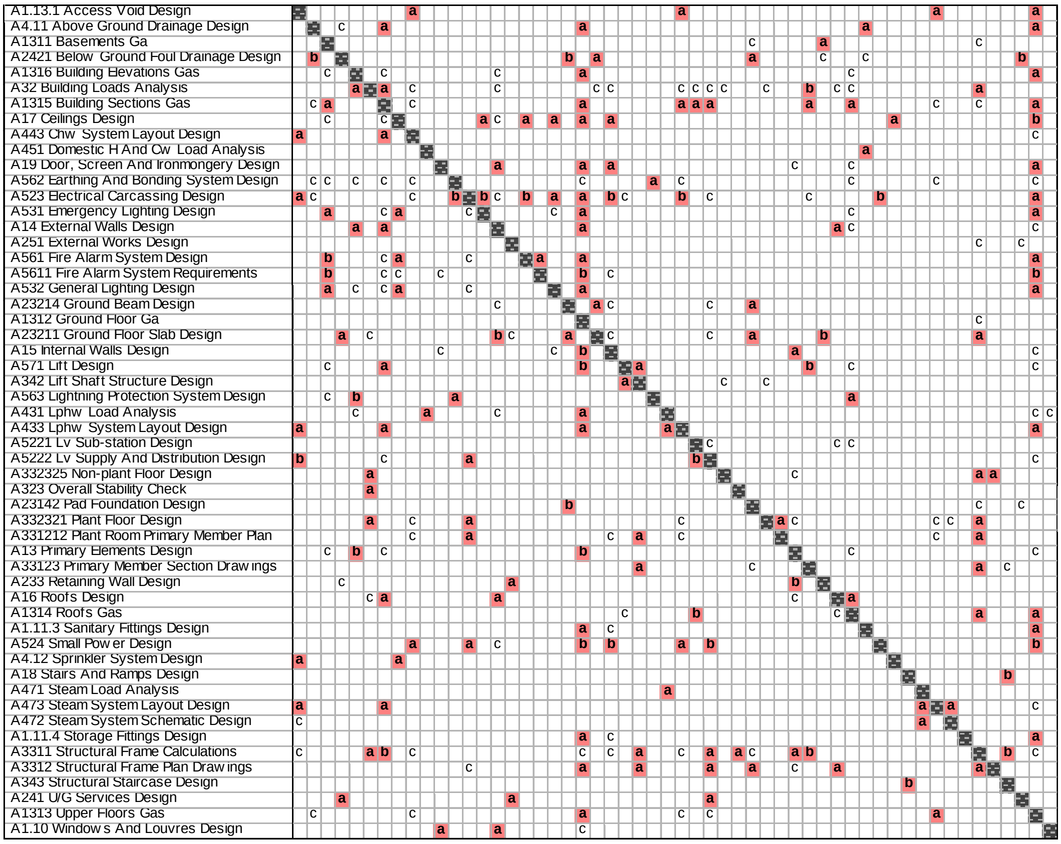

A1.13.1 Access Void Design
A4.11 Above Ground Drainage Design
A1311 Basements Ga
A2421 Below Ground Foul Drainage Design
A1316 Building Elevations Gas
A32 Building Loads Analysis
A1315 Building Sections Gas
A17 Ceilings Design
A443 Chw System Layout Design
A451 Domestic H And Cw Load Analysis
A19 Door, Screen And Ironmongery Design
A562 Earthing And Bonding System Design
A523 Electrical Carcassing Design
A531 Emergency Lighting Design
A14 External Walls Design
A251 External Works Design
A561 Fire Alarm System Design
A5611 Fire Alarm System Requirements
A532 General Lighting Design
A23214 Ground Beam Design
A1312 Ground Floor Ga
A23211 Ground Floor Slab Design
A15 Internal Walls Design
A571 Lift Design
A342 Lift Shaft Structure Design
A563 Lightning Protection System Design
A431 Lphw Load Analysis
A433 Lphw System Layout Design
A5221 Lv Sub-station Design
A5222 Lv Supply And Distribution Design
A332325 Non-plant Floor Design
A323 Overall Stability Check
A23142 Pad Foundation Design
A332321 Plant Floor Design
A331212 Plant Room Primary Member Plan
A13 Primary Elements Design
A33123 Primary Member Section Draw ings
A233 Retaining Wall Design
A16 Roofs Design
A1314 Roofs Gas
A1.11.3 Sanitary Fittings Design
A524 Small Pow er Design
A4.12 Sprinkler System Design
A18 Stairs And Ramps Design
A471 Steam Load Analysis
A473 Steam System Layout Design
A472 Steam System Schematic Design
A1.11.4 Storage Fittings Design
A3311 Structural Frame Calculations
A3312 Structural Frame Plan Draw ings
A343 Structural Staircase Design
A241 U/G Services Design
A1313 Upper Floors Gas
A1.10 Window s And Louvres Design
a
b
c

Fig. 6 A matrix from one of the case studies, produced with PSM

| ### | Label |
|---|---|
| 1 | sitesurv |
| 2 | siteinv |
| 10 | slabdes |
| 4 | pierdes |
| 7 | deckdra |
| 9 | deakmsw |
| 3 | foundat |
| 14 | superst |
| 11 | decklsd |
| 5 | abutmen |
| 13 | windloa |
| 15 | footpat |
| 6 | bearing |
| 8 | deckmat |
| 12 | deckter |
| 17 | signpos |
| 16 | safetyg |

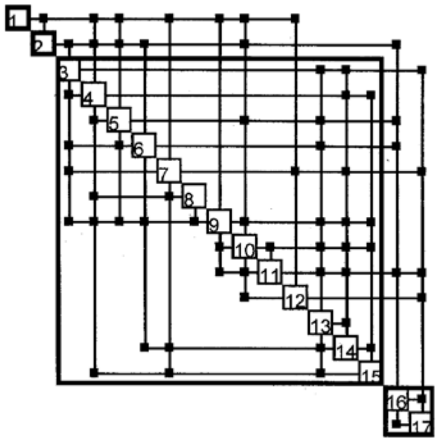
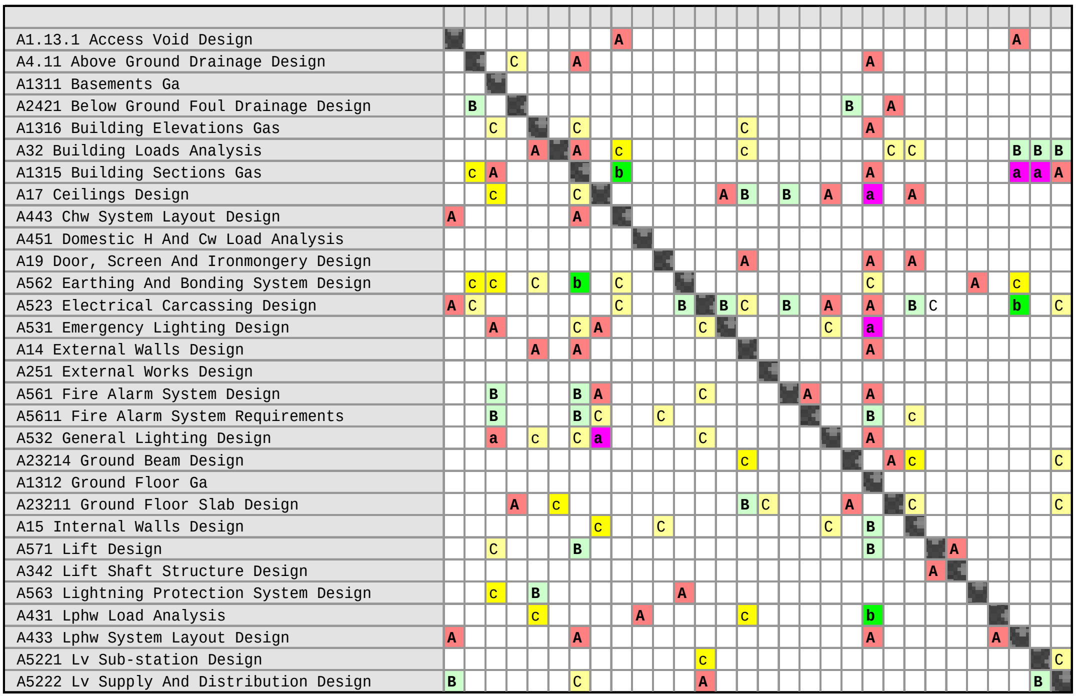
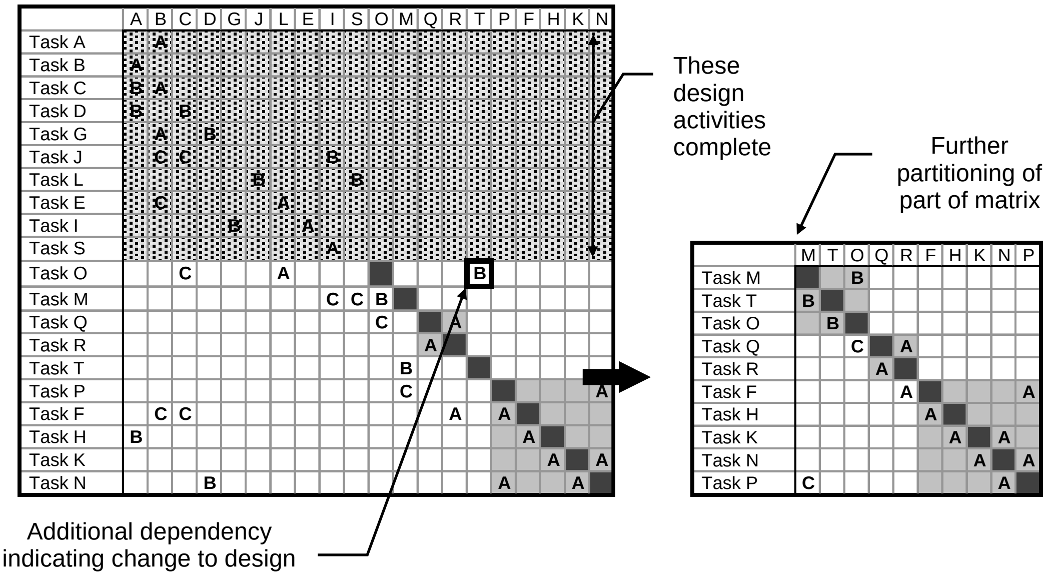


Fig. 7 A matrix produced with DeMAID/GA



| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A1.13.1 Access Void Design | ■ | | | | | | | A | | | | | | | | | | | | | | | | | | | A | | |
| A4.11 Above Ground Drainage Design | | ■ | | C | | | A | | | | | | | | | | | | | A | | | | | | | | | |
| A1311 Basements Ga | | | ■ | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A2421 Below Ground Foul Drainage Design | | B | | ■ | | | | | | | | | | | | | | | B | | A | | | | | | | | |
| A1316 Building Elevations Gas | | | C | | ■ | | C | | | | | | | C | | | | | | A | | | | | | | | | |
| A32 Building Loads Analysis | | | | | A | ■ | A | | c | | | | | c | | | | | | | C | C | | | | | B | B | B |
| A1315 Building Sections Gas | | c | A | | | | ■ | | b | | | | | | | | | | | A | | | | | | | a | a | A |
| A17 Ceilings Design | | | c | | | | C | ■ | | | | | A | B | | B | | A | | a | | A | | | | | | | |
| A443 Chw System Layout Design | A | | | | | | A | | ■ | | | | | | | | | | | | | | | | | | | | |
| A451 Domestic H And Cw Load Analysis | | | | | | | | | | ■ | | | | | | | | | | | | | | | | | | | |
| A19 Door, Screen And Ironmongery Design | | | | | | | | | | | ■ | | | A | | | | | | A | | A | | | | | | | |
| A562 Earthing And Bonding System Design | | c | c | | c | | b | | C | | | ■ | | | | | | | | C | | | | | A | | c | | |
| A523 Electrical Carcassing Design | A | C | | | | | | | C | | | B | ■ | B | C | | B | | A | A | | B | C | | | | b | | C |
| A531 Emergency Lighting Design | | | A | | | | C | A | | | | | C | ■ | | | | | C | a | | | | | | | | | |
| A14 External Walls Design | | | | | A | | A | | | | | | | | ■ | | | | | A | | | | | | | | | |
| A251 External Works Design | | | | | | | | | | | | | | | | ■ | | | | | | | | | | | | | |
| A561 Fire Alarm System Design | | | B | | | | B | A | | | | | C | | | | ■ | A | | A | | | | | | | | | |
| A5611 Fire Alarm System Requirements | | | B | | | | B | C | | | C | | | | | | | ■ | | B | | c | | | | | | | |
| A532 General Lighting Design | | | a | | c | | C | a | | | | | C | | | | | | ■ | A | | | | | | | | | |
| A23214 Ground Beam Design | | | | | | | | | | | | | | | c | | | | ■ | | A | c | | | | | | | C |
| A1312 Ground Floor Ga | | | | | | | | | | | | | | | | | | | | ■ | | | | | | | | | |
| A23211 Ground Floor Slab Design | | | | A | | c | | | | | | | | | B | C | | | A | | ■ | C | | | | | | | C |
| A15 Internal Walls Design | | | | | | | | c | | | C | | | | | | | | C | B | | ■ | | | | | | | |
| A571 Lift Design | | | C | | | | B | | | | | | | | | | | | | B | | | ■ | A | | | | | |
| A342 Lift Shaft Structure Design | | | | | | | | | | | | | | | | | | | | | | | A | ■ | | | | | |
| A563 Lightning Protection System Design | | | c | | B | | | | | | | A | | | | | | | | | | | | | ■ | | | | |
| A431 Lphw Load Analysis | | | | | c | | | | | A | | | | | c | | | | | b | | | | | | ■ | | | |
| A433 Lphw System Layout Design | A | | | | | | A | | | | | | | | | | | | | A | | | | | | A | ■ | | |
| A5221 Lv Sub-station Design | | | | | | | | | | | | | c | | | | | | | | | | | | | | | ■ | C |
| A5222 Lv Supply And Distribution Design | B | | | | | | C | | | | | | A | | | | | | | | | | | | | | | B | ■ |

Fig. 8  A matrix produced using AMMP



| | A | B | C | D | G | J | L | E | I | S | O | M | Q | R | T | P | F | H | K | N |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Task A | | A | | | | | | | | | | | | | | | | | | |
| Task B | A | | | | | | | | | | | | | | | | | | | |
| Task C | B | A | | | | | | | | | | | | | | | | | | |
| Task D | B | | B | | | | | | | | | | | | | | | | | |
| Task G | | A | | B | | | | | | | | | | | | | | | | |
| Task J | | C | C | | | | | | B | | | | | | | | | | | |
| Task L | | | | | | B | | | | B | | | | | | | | | | |
| Task E | | C | | | | | A | | | | | | | | | | | | | |
| Task I | | | | | B | | | A | | | | | | | | | | | | |
| Task S | | | | | | | | | A | | | | | | | | | | | |
| Task O | | | C | | | | A | | | | ■ | | | | B | | | | | |
| Task M | | | | | | | | | C | C | B | ■ | | | | | | | | |
| Task Q | | | | | | | | | | | C | | ■ | A | | | | | | |
| Task R | | | | | | | | | | | | | A | ■ | | | | | | |
| Task T | | | | | | | | | | | | B | | | ■ | | | | | |
| Task P | | | | | | | | | | | | C | | | | ■ | | | | A |
| Task F | | C | C | | | | | | | | | | | A | | A | ■ | | | |
| Task H | B | | | | | | | | | | | | | | | | A | ■ | | |
| Task K | | | | | | | | | | | | | | | | | | A | ■ | A |
| Task N | | | | B | | | | | | | | | | | | A | | | A | ■ |

These design activities complete

Further partitioning of part of matrix

| | M | T | O | Q | R | F | H | K | N | P |
|---|---|---|---|---|---|---|---|---|---|---|
| Task M | ■ | | B | | | | | | | |
| Task T | B | ■ | | | | | | | | |
| Task O | | B | ■ | | | | | | | |
| Task Q | | | C | ■ | A | | | | | |
| Task R | | | | A | ■ | | | | | |
| Task F | | | | | A | ■ | | | | A |
| Task H | | | | | | A | ■ | | | |
| Task K | | | | | | | A | ■ | A | |
| Task N | | | | | | | | A | ■ | A |
| Task P | C | | | | | | | | A | ■ |

Additional dependency indicating change to design

Fig. 9  Partitioning part of a matrix

| **Project** | | **A** | **B** | **C** | **D** |
|---|---|---|---|---|---|
| Description | | Pharmaceutical laboratory | Railway terminal | Office development | Hospital |
| No. of design tasks | | 410 | 357 | 346 | 789 |
| No. of information dependencies | | 2,406 | 2,804 | 2,656 | 10,015 |
| No. of iterative loops after tearing | | 14 | 14 | 7 | 19 |
| Proportion of tasks in loops | | 29% | 18% | 18% | 29% |
| Hours to generate | Model | 16 | 28 | 12 | 32 |
| | Matrix | 20 | 20 | 16 | 40 |
| | Programme | 28 | 24 | 24 | 40 |

Table 1  Results of applying a DSM tool to building projects