The 9th International Symposium on Automation and Robotics in Construction June 3-5, 1992 Tokyo, Japan

CONSTRUCTION ROBOTICS MANAGEMENT WITH A PERSONAL COMPUTER

Mirosław J. Skibniewski

Division of Building, Construction and Engineering
Commonwealth Scientific and Industrial Research Organisation
P.O. Box 56, Graham Rd.
Highett, Victoria 3190, Australia

on leave from

Division of Construction Engineering and Management
School of Civil Engineering
Purdue University
West Lafayette, Indiana 47907, U.S.A.

Kinya Tamaki
Systems Science Institute
Waseda University
3-4-1 Okubo, Shinjuku-ku
Tokyo 169, Japan

Jose E. Thomaz
School of Civil Engineering
Purdue University
West Lafayette, Indiana 47907, U.S.A.

ABSTRACT

The Construction Robotic Equipment Management System (CREMS) has been developed as a response to the need by one of the best known Japanese construction and engineering firms to effectively manage diverse robots on future construction sites. This paper outlines a generic decision logic for comparison of conventional work methods and available robotics for the performance of construction tasks, as well as the robot application logistics. Modules comprising the system and the inter-relationships between the system modules are presented. An example Macintosh personal computer application of CREMS to a concrete floor finishing robot evaluation and application logistics is presented.

1. INTRODUCTION

Robotic applications to the execution of construction tasks pose significant challenges to the management of these tasks performed on jobsites. An even greater challenge comes from the fact that several leading Japanese construction firms already have entire fleets of diverse construction robots available for implementation at their jobsites. A summary description of currently implemented systems can be found in (Skibniewski and Russell 1989).

Despite a number of advantages over traditional methods of performing construction tasks, robots are currently in short supply in comparison with other construction equipment. Thus, they should be regarded as a scarce resource and their use should be maximized to their full operating potential. With maximized robot utilization on as many construction projects as possible in the contractor's portfolio, economic benefits of robot use can be easier to attain. Consequently, robot development costs can be recovered faster and robot use can spread to other applications and types of construction tasks.

The Construction Robotic Equipment Management System (CREMS) was conceived as a response to this need. The principal objective of CREMS is to provide a comprehensive construction robot management capability for construction field personnel. CREMS is intended as an aid in decision-making regarding the implementation of robots on the jobsite. Such a system should prove highly desirable when managing a fleet of diverse single-purpose robots and increase the application efficiency of robots available to a construction company.

CREMS analyzes the available data on the construction task considered for robot performance, assesses the capability of the available robot, computes the task performance complexity and efficiency for both performance options, compares the economic advantages of either choice, and determines optimal scheduling of a given robot throughout all relevant project sites in the current firm's portfolio.

CREMS application implies that the types of robotics under consideration for the performance of construction tasks are regarded as "fixed quantity." In other words, the design configuration of all robots in the company fleet is assumed as fixed, the capabilities, productivity rates and reliability of the robotic equipment are known from a sufficient amount of past experience with these robots on projects similar to the ones currently in the company portfolio. This is clearly a limitation of the presented system prototype, since most of the available robots and their designs are still in prototypical stages, and most construction firms in the possession of robotic equipment have limited experience with their use on commercial job sites. As more experience is gathered with the robotics hardware, and a better knowledge of their performance capabilities is compiled, the utility of CREMS for management decision support in the optimal assignment of robots to job tasks will become more apparent.

2. CREMS ARCHITECTURE

CREMS, as shown in Figure 1, consists of four basic modules: Construction Task Analysis Module (CTAM), Robot Capability Analysis Module (RCAM), Robot Economics Evaluation Module (REEM), and Robot Implementation Logistics Module (RILM). Following is a brief description of each module.

Construction Task Analysis Module (CTAM).- This module acquires and analyzes all pertinent data regarding a given construction task. The data analyzed include manpower and productivity requirements, achievable work quality, and characteristics of the work environment. The information derived is both a qualitative and quantitative evaluation of task suitability for robotic and non-robotic performance.

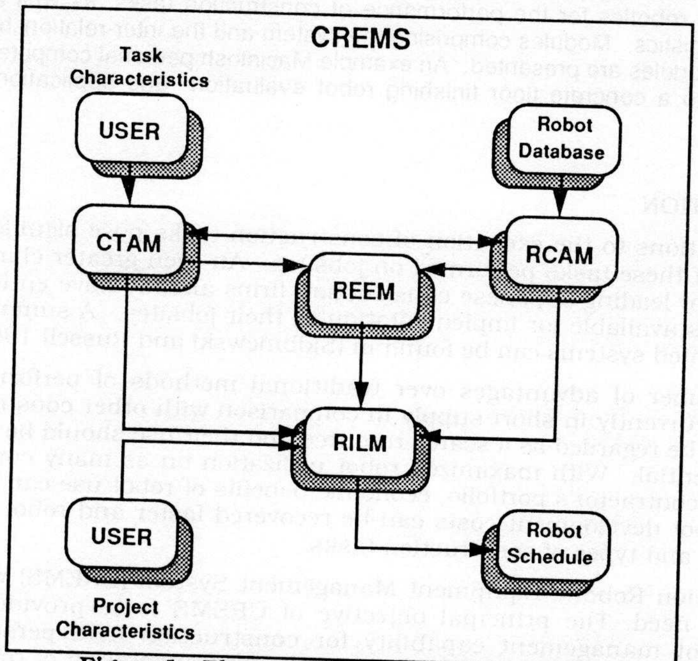


Figure 1. The Principal Modules of CREMS

Robot Capability Analysis Module (RCAM).- This module examines the capabilities of a given robot or group of robots for the performance of a construction task(s). The data analyzed include robot weight, power supply, payload, work envelop, available tools, and control and sensory systems.

Robot Economics Evaluation Module (REEM).- This module performs a detailed comparative analysis of non-robotic versus robot jobsite work alternatives relevant for the considered jobsite location. Factors for analysis include task performance cost with non-robotic work techniques, operational costs of robotic equipment, resulting labor and time savings, and quality improvements.

Robot Implementation Logistics Module (RILM).- This module performs robotic equipment scheduling functions to ensure optimal use throughout the company projects. These functions include robot work time determination, tooling and manpower allocation, robot maintenance schedules, robot site set-up and disassembly, and transportation schedules. During the scheduling and resource leveling procedures, the robots are treated simply as another project resource are and subject to similar management approach as other conventional construction equipment.

3. ROBOT APPLICATION LOGISTICS

RILM is divided into seven major procedural blocks, listed in Figure 2.

Procedure 1: Determines Number of Robots
Procedure 2: Compiles Project Database
Procedure 3: Selects Preference Rule and List Selected Projects
Procedure 4: Performs Robot Transportation Analysis
Procedure 5: Preforms Short Task Cycle (STC) Project Scheduling
Procedure 6: Preforms Long Task Cycle (LTC) Project Scheduling
Procedure 7: Reports Logistic Scheduling Results
Output: Robots Implementation Logistic Scheduling

Figure 2: RILM Main Procedures

RILM develops a Downhill chart or a Gantt Chart, which contains the original data collected by the computer from the Project Database Stack. The Downhill Chart is used to assign task-projects to different phases. The phases in the chart are represented by the different horizontal levels and are classified in either robot and non-robot phases. The number of robot phases always equals the number of robots available for the projects in consideration.

The five most conceptually innovative features in RILM are shown in Figure 3. They are described in the following paragraphs.

1. Project DatabaseStack
2. Logistic Scheduling for Multiple Numbers of Robots
3. Task Cycle: Short and Long Task-Cycle Project
4. Robot Transportation Analysis
5. Evaluation of Logistic Scheduling Results

Figure 3: Features of RILM

Project Database stack stores information necessary for the analysis of each project. This information includes robot cost and non-robot cost for a certain activity and other logistic scheduling data.

The current version of RILM deals with multiple numbers of robots. Detailed algorithms and software for the multiple several robots have been developed (Skibniewski and Russell 1991).

Task Cycle is defined as the duration in days of the task-projects involved in a project. One task cycle is composed of a task-project (working period) and an idle time period. Short Task Cycle (STC) projects are the ones where the task cycle is small (e.g., less or equal to 5 days) and the robot has to remain on the same project site even during idle time. Only short task cycle projects can be subjects to crash time performance.

Long Task Cycle (LTC) projects are projects with longer task cycles, where the robot can be transported between construction sites and between task-projects. The last task cycle of a project always contains only the task-project (i.e., no idle time).

The scope of robot transportation assumes the distance travelled within one-day or half-day by truck (for example, within the Kanto area in Tokyo). The possible options for transportation include the normal transportation during daytime and overnight transportation in the case requiring quick transportation between two project sites. Therefore, in case of normal transportation, one day is left between the last task-project of a project and the first task-project of the next project in the same robot-phase.

At the end of RILM analysis, the final results of the logistic scheduling are evaluated with statistical calculations and graphic figures. RILM allows the user to feed back into the analysis procedure if the obtained scheduling results are not acceptable to him. RILM also includes a reporting function for the scheduling process results.

4. LOGISTICS SOFTWARE DEVELOPMENT

"Hypermedia" recently emerged as one of the most popular technologies for representing and processing complex information. Some background on the current state of this technology can be found in (Begoray 1990). HyperCard™ is an example hypermedia software environment in which the CREMS architecture is being implemented. Module programs have been implemented with the use of the HyperTalk™ scripts, a programming language within HyperCard. The developed system is operational on Macintosh IIe through Macintosh IIfx microcomputers. Relevant project-specific inputs to the system and a variety of arithmetic computations are required, for which functions embedded within HyperCard are used. Graphic capabilities of HyperCard allow the presentation of relevant data to a CREMS user. CREMS determines whether a robot is available to perform a given construction task, whether the robot use is economically justifiable, and in the case when multiple projects warrant the use of the robot, selection of the project for deployment. A number of features are provided to aid the user during a CREMS consultation. The system, as currently implemented, identifies the task characteristics, selects the robot available to perform the task, and evaluates the economic attractiveness of robot use. An algorithm for making optimal robot assignments when multiple construction project sites are considered has also been developed and integrated within CREMS.

RILM is a fairly complex module from the programming standpoint. Requirements included not only a friendly interface, consistent with the other modules, but also smooth integration and the incorporation of some computationaly intensive tasks for which an interpreted language like Hypercard would prove excessively slow.

In order to accommodate these conditions without significant departure from the programing organization of the other CREMS modules developed with Hypercard, it was decided that only the routines performing most time demanding tasks would be written in conventional programming languages.

As a result, RILM is written in Hypercard and contains routines written in Quick Basic and Think Pascal. These routines were used specifically for the STC (Short Task Cycle) and LTC (Long Task Cycle) analyses, as well as for the creation of the final report. Pascal was chosen for its clarity and self documentation properties, as well as its good support for data structures. These proved very useful in keeping the algorithm code easy to understand (Figure 6-6) and should make future maintenance of the program considerably easier than if it had been written in Hypercard. Think Pascal also provides excellent execution speed once compiled. These languages are discussed in more detail in the following paragraphs.

CREMS is developed with Hypertalk 2.0, an interpreted language for Macintosh. The major feature of Hypercard is the simplicity with which interfaces can be designed. The screens are functionally identical to cards, in which buttons, data fields, graphics and text are placed by simple "cut," "paste" and "drag" actions with the mouse. Hypercard interface creation tools are completely "Object Oriented". Procedures can be encapsulated in buttons and output fields, creating a very good environment for prototyping. Hypercard 2.0 has even better capabilities than former versions. These new features include the capability to use resizable cards that use up to the full screen, the capability to display more than one stack (collection of cards) at the same time on separate windows, a more complete programming language, better debuging capabilities, and faster execution through the generation of pseudo-compiled code.

Under certain circumstances, specific tasks in the RILM module required the creation of charts. Although Hypercard 2.0 does include some new functions for the creation of charts, it was found that execution was usually slow. Thus whenever this need arose, *Microsoft* Quick Basic Compiler version 1.0b was used. This compiler incorporates a fairly powerful dialect of Basic and fully supports the Macintosh toolbox routines. The latest version of the compiler works well under all system software versions up to 6.7. It seems that *Microsoft* will be releasing a new version that will be compatible with the new System 7 just released. Applications compiled with the current version of the compiler do run under System 7 but the compiler itself does not.

In the RILM module there are algorithms that require substantially more speed than Hypercard is capable of. Furthermore, the number of variables in these algorithms is very large and the use of records simplifies the readability of the program.

These special requirements led to the implementation of these routines using *Symantec's* Think-Pascal version 3.05, a popular Pascal implementation for the Macintosh computer. This implementation closely follows the ANSI Pascal specifications, with enough extensions to cover the Macintosh *Toolbox*. It also contains object-oriented extensions, since the Macintosh computer operating system itself is built upon this concept. None of these extensions are used in the algorithms, though, which should make them easily portable to other platforms.

```
    projecttype = record
      number: integer;              {project number label}
      priority: integer;            {project priority}
      startdate: integer;
      latestartdate: integer;
      enddate: integer;
      lateenddate: integer;
      reductiontime: integer;       {reduction time}
      slacktime: integer;
      fixedtermination: integer;    {yes/no}
      robotphase: boolean;          {yes /no if allocated to robot phase}
      phasenumber: integer;         {number of phase the robot is allocated to}
      ntcycles: integer;            {number of task cycles}
      dayspertc: integer;           {days per tasc cycle}
      taskcyclephase: array[1..30] of integer;
      taskcycle: array[1..60] of tasktype;
    end;

     arrproject = array[1..10] of projecttype;
     inte10 = array[1..10] of integer;
     mattype = array[1..10, 1..365] of integer;  { matrix of phase x days for}
                                                 { task project allocation}

     var

     filin: text;
     allocation: mattype;
     numberofrobots: integer;
     numberofprojects: integer;
     project: arrproject;
     pidentified: integer;      {project to be placed next}
     pselected: integer;        {project already placed and interfering with the identified
project}
```

Figure 4: Data Structures for Self Documentation and Algorithm Clarity

The media selected for the exchange of information within the different submodules and external routines consist of data files. The interaction of files follows the arrow lines between the blocks. The file blocks are Hypercard files, shared files, Quickbasic files and Think Pascal files.

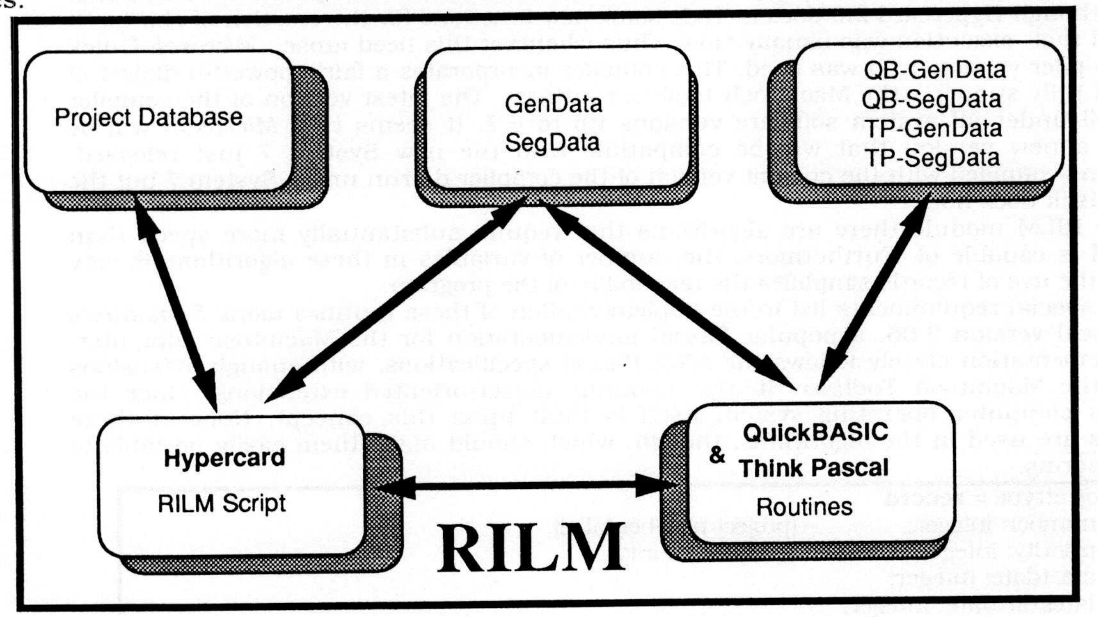
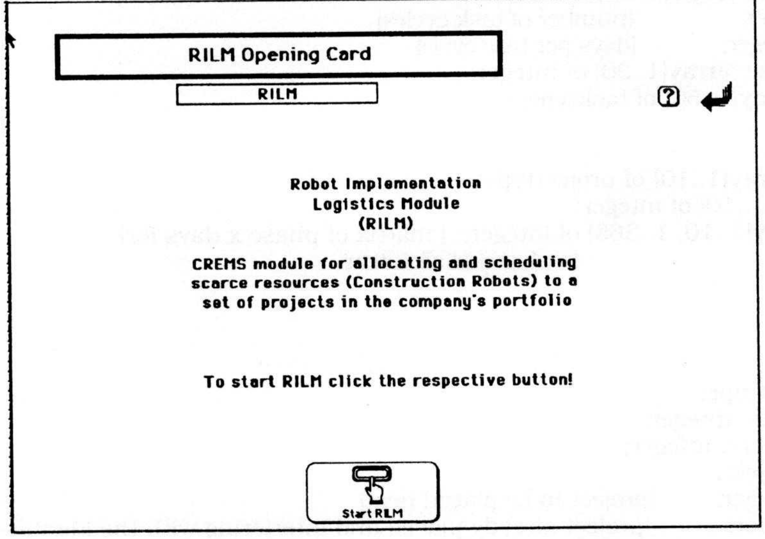


Figure 5: RILM Configuration and Files Interaction

The main program was written in Hypercard, which is used to present the results on the screen and to call the QuikBASIC and Think Pascal programs. The programming within Hypercard involves only the opening and closing of files and the flow of data through the module. Figure 6 presents the opening card of the robot implementation logistics routine.


RILM Opening Card

RILM

Robot Implementation
Logistics Module
(RILM)

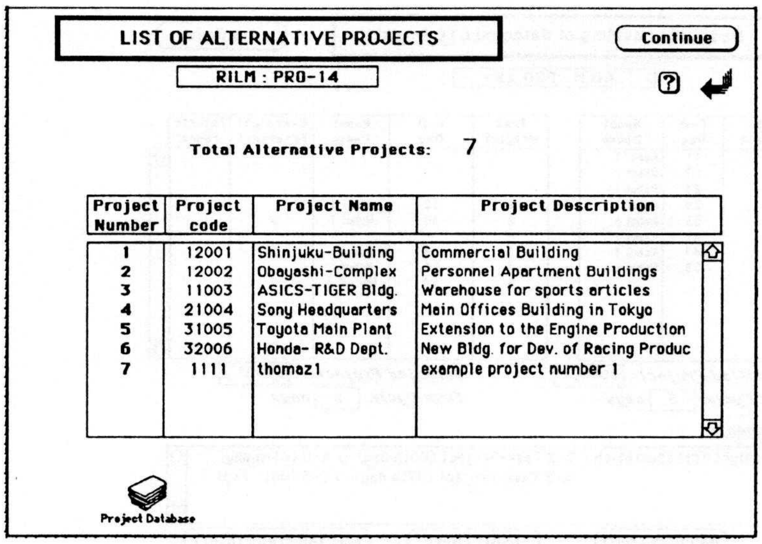
CREMS module for allocating and scheduling scarce resources (Construction Robots) to a set of projects in the company's portfolio

To start RILM click the respective button!

Start RILM

Figure 6: Starting the Robot Implementation Logistic Scheduling

The routine requires appropriate information on the projects for which robot implementation is being considered. An example project information input provided from a project portfolio database is presented in Figure 7.


LIST OF ALTERNATIVE PROJECTS

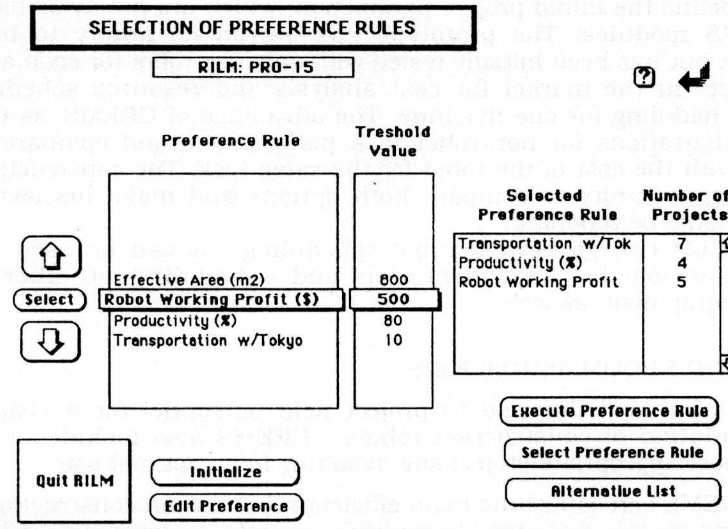
RILM : PRO-14

Total Alternative Projects: 7

Project Number	Project code	Project Name	Project Description
1	12001	Shinjuku-Building	Commercial Building
2	12002	Obayashi-Complex	Personnel Apartment Buildings
3	11003	ASICS-TIGER Bldg.	Warehouse for sports articles
4	21004	Sony Headquarters	Main Offices Building in Tokyo
5	31005	Toyota Main Plant	Extension to the Engine Production
6	32006	Honda- R&D Dept.	New Bldg. for Dev. of Racing Produc
7	1111	thomaz1	example project number 1

Figure 7: List of Alternative Projects for Robot Implementation

A number of optimization criteria can be utilized in the robot implementation scheduling process. These criteria are frequently related to the maximization of the net profit to the company from the use of robots in the fleet. An example program input related to such criteria is presented in Figure 8.


SELECTION OF PREFERENCE RULES

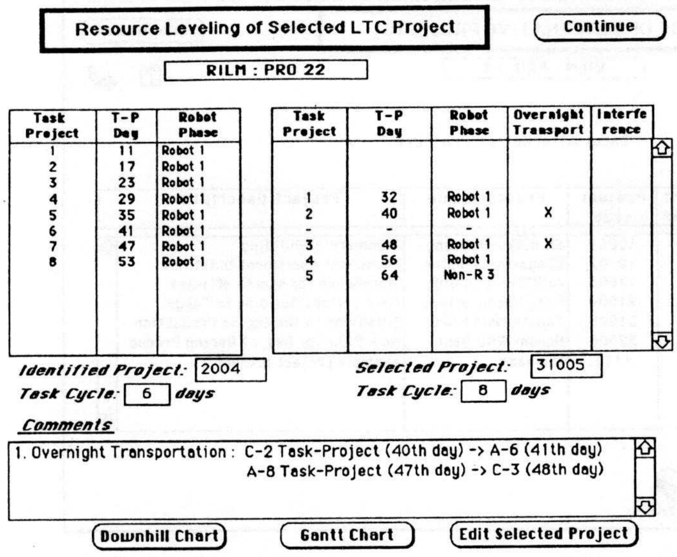
RILM: PRO-15

Preference Rule	Treshold Value
Effective Area (m2)	800
Robot Working Profit ($)	500
Productivity (%)	80
Transportation w/Tokyo	10

Selected Preference Rule	Number of Projects
Transportation w/Tok	7
Productivity (%)	4
Robot Working Profit	5

Figure 8: Selected Robot Implementation Optimization Criteria

The program output is provided in the form of a robot application schedule, from which information on robot assignment is communicated to the project management and construction robot equipment dispatcher. In addition, robot resource leveling is performed to assure the desired distribution of robots to jobsites where their work is technologically feasible in regard to constraints indirectly related to the task for which the robot is intended. An example resource leveling output is provided in Figure 9.


Resource Leveling of Selected LTC Project

Continue

RILM : PRO 22

Task Project	T-P Day	Robot Phase
1	11	Robot 1
2	17	Robot 1
3	23	Robot 1
4	29	Robot 1
5	35	Robot 1
6	41	Robot 1
7	47	Robot 1
8	53	Robot 1

Task Project	T-P Day	Robot Phase	Overnight Transport	Interfe rence
1	32	Robot 1		
2	40	Robot 1	X	
-	-	-		
3	48	Robot 1	X	
4	56	Robot 1		
5	64	Non-R 3		

Identified Project: 2004

Task Cycle: 6 *days*

Selected Project: 31005

Task Cycle: 8 *days*

Comments

1. Overnight Transportation : C-2 Task-Project (40th day) -> A-6 (41th day)
A-8 Task-Project (47th day) -> C-3 (48th day)

Figure 9: Resource Leveling of a Selected Project

5. BENEFITS OF RILM

RILM performs project resource leveling independent of the task and the robot, even though the user has to define the initial project parameters, which are not available from the other first three CREMS modules. The prototype has proven its ability to handle the parameters of any project but has been initially tested with only one robot for each analysis.

The different products in the market for cost analysis and resource scheduling only analyze the costs and scheduling for one machine. The advantage of CREMS is that it can analyze many task configurations for non-robot task performance and compare the most favorable configuration with the cost of the robot for the same task. The construction project manager will have the opportunity to compare both options and make his own decision whether the given task should be robotized.

The application of RILM can go beyond robot scheduling. It can be used in today's construction projects for logistic implementation and scheduling of other valuable construction and heavy equipment, as well.

6. CONCLUSIONS AND RECOMMENDATIONS

CREMS is intended as a real time aid to project field personnel for decision-making relative to the implementation of construction robots. CREMS also includes a facility to deploy the robot to the most appropriate project site, resulting in its optimal use.

The application of CREMS can contribute to an efficient robot use on construction jobsites, to planning and assigning robotic resources to multiple projects, in promoting robot use on construction sites, and to making an investment in robot development or purchase a worthwhile endeavor.

As described in the paper, the initial prototype of CREMS has been developed for a Macintosh II personal computer within the Hypercard™ programming environment. Hypercard is a comprehensive database programming environment allowing efficient integration of numerical, semantic, and graphical data relevant to the description of construction project parameters, as well as robot capability and performance information associated with project task execution.

The current prototype version of CREMS is being upgraded to be more "user-friendly," based on detailed feedback from program testing procedures In later versions, the user will not need to type any text into the program and will thus eliminate possible typing errors. This will also eliminate the need for memory and time-consuming error detecting scripts. Also, future versions of CREMS may use direct data entry from a file or a voice digitizer. It is also desirable to investigate interfacing CAD (digitized drawings) with this system to recommend potential use of robots on construction jobsites. Such a comprehensive system would link economic evaluation with the scheduling function. Concepts such as risk analysis and probabilistic modeling can be used to incorporate uncertainty in data present within the REEM and RILM modules. The current RILM implementation also includes the integration of Hypercard and QuickBasic programming languages for the efficiency of arithmetic computations on project data.

The experience with CREMS application will also serve as feedback to construction robotic system designers. The information of particular value as feedback includes the sensitivity of robot functional and economic advantage to parameters such as robot dimensions, weight, motion and task performance speed, energy consumption, setup, dismantling and transportation requirements, etc.

CREMS, upon completion, is intended for the field management of a minimum of 15 functionally diverse types of construction robotics, developed in-house at the corporate technical research and development institute, on company project sites worldwide.

ACKNOWLEDGEMENTS

Support for this research from the Obayashi Corporation as well as from the National Science Foundation Presidential Young Investigator Award No. MSM-8657142 to Professor M.J. Skibniewski is gratefully acknowledged.

REFERENCES

Begoray, J. A. (1990). "An Introduction to Hypermedia Issues, Systems and Application Areas," International Journal of Man-Machine Studies, Vol. 33, No. 2, August, pp. 121-147.

Bhatti, M. A. (1988). "Developing Engineering Design Software Using HyperCard," Microcomputers in Civil Engineering, (3) pp. 111-126.

Cornejo, C., Tamaki, K., Posey, J., Skibniewski, M. (1991). "Computer Assisted System for Construction Robot Implementation Logistics," Proceedings of the ASCE 7th Conference on Computing in Civil Engineering, Washington, D.C., May, pp 432-441.

Goodman, D. (1988). The Complete HyperCard Handbook, Bantam Brooks, New York, NY.

HyperCard™ User's Guide, (1989). Apple Computer, Inc., Cupertino, CA.

Nunnaly, S. (1977). Managing Construction Equipment, Prentice-Hall, Englewood Cliffs, NJ.

Oppenheim, I., Skibniewski, M.*: *"Robots in Construction,"* in "International Encyclopedia of Robotics," John Wiley & Sons, A Wiley-Interscience Publication, Vol. 1, New York, 1988, ISBN 0-471-63512-4, pp. 240-249

Peurifoy, R., Ledbetter, W. (1985). Construction Planning, Equipment and Methods, McGraw-Hill Publishing Co. (Series in Construction Engineering and Project Management), New York, N.Y.

Russell, J. and Skibniewski, M. (1991). "An Ergonomic Analysis Framework for Construction Tasks," Vol. 8, Construction Management and Economics, Chapman and Hall Publishers, London, England, 1990, pp. 329-338.

Russell, J., Skibniewski, M., and Vanegas, J. (1990). "Framework for Construction Robot Fleet Management System," ASCE Journal of Construction Engineering and Management, Vol. 116, No. 3, September, pp. 448-462.

Touran, A., Taher, K. (1988). "Optimum Fleet Size Determination by Queueing and Simulation," Construction Management and Economics, Vol. 6, E. & F.N. Spon Publishers, London, England, pp. 295-306.

Skibniewski, M. (1988a). *Robotics in Civil Engineering*, Van Nostrand Reinhold Publishing Co., New York, New York.

Skibniewski, M. J. (1988). "Framework for Decision-Making on Implementing Robotics in Construction," ASCE Journal of Computing in Civil Engineering, Vol. 2, No. 2, April, pp. 188-201.

Skibniewski, M: "*Robot Implementation Issues for the Construction Industry*," in "Human Robot Interaction," Rahimi, M. and Karwowski, W., eds., Taylor & Francis Publishers, London, U.K., - Washington, D.C., 1992, ISBN 0-85066-809-3, pp. 347-366

Skibniewski, M. J. and Hendrickson, C. (1988). "Analysis of Robotic Surface Finishing Work on the Construction Site," ASCE Journal of Construction Engineering and Management, Vol. 115, No. 2, March pp. 53-68.

Skibniewski, M. J. and Russell, J. S. (1989). "Robotic Applications To Construction," AACE Journal of Cost Engineering, Vol. 31, No. 6, June, pp. 10-18.

Skibniewski, M.*:, Russell, J. (1991): "Construction Robot Fleet Management System Prototype," ASCE *Journal of Computing in Civil Engineering*, Vol 5, No. 4, October, pp. 444-463

Skibniewski, M., Tamaki, K., and Russell, J. (1990). "Construction Robot Implementation Logistics," Proceedings, Vol. 2, 7th International Symposium on Automation and Robotics in Construction, Bristol, England, June., pp. 543-555.

Skibniewski, M. J., Vanegas, J. A., and Russell, J. S. (1989). "Construction Robotic Equipment Management System (CREMS)," Proceedings, *Sixth International Symposium on Automation and Robotics in Construction*, Sponsored by the Construction Industry Institute, San Francisco, California, pp. 404-411.