Mobile Model-Based Bridge Lifecycle Management Systems

Amin Hammad*

Concordia Institute for Information Systems Engineering

1425 René Lévesque Boulevard, Montréal, Québec, H3G 1T7, Canada

Cheng Zhang, Yongxin Hu & Elaheh Mozaffari

Department of Civil, Building & Environmental Engineering, Concordia University

Abstract: This paper discusses the requirements for developing Mobile Model-based Bridge Lifecycle Management Systems (MMBLMSs). These new systems should link all the information about the lifecycle stages of a bridge (e.g., design, construction, inspection and maintenance) to a 4D model of the bridge incorporating different scales of space and time in order to record events throughout the lifecycle with suitable levels of details (LoDs). In addition, MMBLMSs should support distributed databases and mobile location-based computing by providing user interfaces that could be used on thin clients, such as tablet PCs. A framework of MMBLMSs is described and the basic computational issues for realizing it are discussed including the navigation modes, picking behavior and the LoDs for representing bridge elements and defects. A prototype system developed in Java

* To whom correspondence should be addressed. E-mail: hammad@ciise.concordia.ca
language is used to demonstrate the feasibility of the proposed methodology for realizing these systems.

1 INTRODUCTION

This paper discusses the requirements for developing Mobile Model-based Bridge Lifecycle Management Systems (MMBLMSs). These new systems should link all the information about the lifecycle stages of a bridge (e.g., design, construction, inspection and maintenance) to a 4D model of the bridge incorporating different scales of space and time in order to record events throughout the lifecycle. In addition, MMBLMSs should support distributed databases and mobile Location-Based Computing (LBC) by providing user interfaces that could be used on thin clients, such as PDAs and tablet PCs, equipped with wireless communications and tracking devices, such as Global Positioning System (GPS) receivers.

The paper starts by reviewing conventional Bridge Management Systems (BMSs) and recent trends in 4D models, mobile computing and LBC. This is followed by an analysis of the requirements of the proposed MMBLMSs. Special consideration is given to the spatial and temporal issues, such as the requirements to support navigation, picking behavior and different Levels of Details (LoDs), and to adopt available standards for interoperability. A framework of MMBLMSs is described and the basic computational issues for realizing it are discussed. Then, a prototype system developed in Java language to demonstrate the feasibility of the proposed methodology is discussed in detail including the system architecture and the database and user interface design. A case study of Jacques Cartier Bridge in Montreal is also demonstrated.
2 REVIEW OF BMSs

Bridge lifecycle management aims to perform the management functionalities related to bridges from the conceptual stage to the end of their useful life, through the design, construction, operation and maintenance stages. The major tasks in bridge management are: (1) collection of inventory data, (2) inspection, (3) assessment of condition and strength, (4) repair, strengthening or replacement, and (5) prioritizing the allocation of funds.

BMSs are means of managing information of bridges to support decision making that assures their long-term health and to formulate maintenance programs in line with budgetary constraints and funding limitations. BMSs include four basic components: data storage, cost and deterioration models, optimization and analysis models, and updating functions (Czepiel, 2004; Ryall, 2001). The core part of a BMS is the database which is built up of information obtained from the regular inspection and maintenance activities. Bridge database management includes the collection, updating, integration, and archiving of the following information: (1) bridge general information (location, name, type, load capacity, etc.), (2) design information and physical properties of the elements, (3) inventory data, (4) regular inspection records, (5) condition and strength assessment reports, (6) repair and maintenance records, and (7) cost records.

New approaches in BMSs try to introduce new information technologies to facilitate mobile data collection and manipulation. For example, a system developed by the University of Central Florida for the Florida Department of Transportation (FDOT) (Kuo et al, 1994) consists of both a field and office set up with a pen-based notebook computer used to collect all field inspection data. The Massachusetts Highway Department is using a system called IBIIS to store and manage all of their bridge documents (Leung, 1996). As part of this system, inspectors are equipped with a video camcorder to take videos and pictures, and a notebook computer to enter the
rating data for each bridge and commentary. A more recent, Personal Digital Assistant (PDA)-based field data collection system for bridge inspection is *Inspection On Hand* (IOH) (Trilon, 2004). IOH helps inspectors capture all rating information, commentary and sketches using hand-held, pen-based PDAs, and share data with Pontis bridge management system. In addition, the Digital Hardhat (DHH) is a pen-based computer with special multimedia facility reporting system that allows the field worker to save multimedia information, such as text, sound, video and images, into a database. DHH technology enables dispersed inspectors to communicate information and to collaboratively solve problems using shared multimedia data (Stumpf et al., 1998).

The proposed approach for MMBLMSs presented in this paper makes the first attempt to integrate 4D bridge models with BMSs and to make the resulting information accessible to mobile on-site workers. Although 4D models have already been built to support construction planning and scheduling (Zhang et al., 2000), these models are not integrated with Facilities Management (FM) or Infrastructure Management Systems (IMS). In addition, there is no available system architecture to support interaction with these models in mobile situations.

### 3 REQUIREMENTS OF MMBLMSs

Using mobile and wearable computers in the field under severe working and environmental conditions requires new types of interaction that increase the efficiency and safety of field workers. Research on systems aiming to provide information related to infrastructure at different stages of their lifecycle to mobile workers has been undertaken. Garrett *et al.* (2002) discussed the issues in delivering mobile and wearable computer-aided inspection systems for field users. Sunkpho *et al.* (2002) developed the Mobile Inspection Assistant (MIA) that runs on a wearable computer and delivers a voice recognition-based user interface. They also proposed a framework for
developing field inspection support systems.

Mobility is a basic characteristic of field tasks. The inspector of a bridge has to move most of the time in order to do the job at hand. The inspector walks over, under or around the bridge, or in some cases climbs the bridge. Knowing the location of the inspector with respect to the inspected elements can greatly facilitate the task of data collection by automatically identifying the elements, and potentially specifying the locations of defects on these elements. Present methods of capturing location information using paper or digital maps, pictures, drawings and textual description can lead to ambiguity and errors in interpreting the collected data.

Location-Based Computing (LBC) is an emerging discipline focused on integrating geoinformatics, telecommunications, and mobile computing technologies (Beadle et al., 1997). LBC utilizes geoinformatics technologies, such as Geographic Information Systems (GISs) and tracking methods, such as the GPS, in a distributed real-time mobile computing environment. In LBC, elements and events involved in a specific task are registered according to their locations in a spatial database, and the activities supported by the mobile and wearable computers are aware of these locations using suitable positioning devices. For example, an inspection system based on LBC would allow the bridge inspector to accurately locate the cracks on a predefined 3D model of the bridge in real time without the need for any post-processing of the data.

The first author (Hammad et al., 2004) discussed the concept and requirements of a mobile data collection system for engineering field tasks called *LBC for Infrastructure field tasks* (LBC-Infra) and identified its system architecture based on available technologies and the modes of interaction. This paper builds on the experience gained from the development and testing of LBC-Infra to propose a new methodology for designing future MMBLMSs that will integrate the different information about the lifecycle of a bridge (e.g., construction,
inspection and maintenance schedules) to the 3D model of the bridge, resulting in 4D models. The following paragraphs discuss the main requirements of MMBLMSs. These requirements are based on interviews with bridge managers and on our experience with previous prototypes of LBC-Infra (Hammad et al., 2004).

(1) **4D modeling and spatio-temporal analysis:** 4D models will facilitate spatio-temporal visualization and analysis that are not possible in present BMSs. This integration of space and time will result in the following advantages: (1) Visualizing different types of data, e.g., displaying the changes in a bridge 3D model at a specific time or during a specific period of its lifecycle; (2) Providing a user-friendly interface which can reduce the data input errors; (3) Facilitating data sharing; and (4) Improving the efficiency of database management. 4D visualization can be understood more quickly and completely than the traditional construction management tools (Fischer, 2001). The Stageworks (Stageworks, 2005) system developed by Bechtel has proved that 4D visualization is helpful during construction. The Navigator software also applies 3D model review, animation and 4D simulation (Bentley, 2005). This requirement is the first step towards future 5D or nD concepts for bridge management, which can incorporate other factors to the model, such as cost, to achieve more comprehensive data integration.

Spatio-temporal analysis is the process of extracting or creating new information about a set of geometric or geographic features at a certain point of time. This type of analysis is useful for evaluating the suitability of a certain location, such as problems in site layout planning, or for predicting spatial conflicts, such as conflicts between workspaces (Akinci et al., 2002). Workspace analysis aims to create different types of workspaces for crew, equipment, and other required spaces in the work site, to detect conflicts between these workspaces, and then to resolve these conflicts.
(2) **Lifecycle data integration**: A uniform bridge inspection reporting system is essential to evaluate the condition of a structure correctly and efficiently, and to establish maintenance priorities. The results of an inspection must be accurately and fully recorded so that a complete history of the structure is available at any time. If available, all of the design information such as drawings, design calculations, soil investigation reports, etc. should be used to help at the inspection and maintenance stages (Itoh et al, 1997). Different types of inspections (inventory, routine, defect and in-depth inspections) allow the bridge owner to establish appropriate inspection levels consistent with the inspection frequency and the type of structure and details. On the other hand, for practical purposes, it is common to subdivide the inspection of a bridge into its main constituent parts, namely the inspection of the superstructure, substructure and foundations, and then to further subdivide these into their separate elements. Condition ratings assigned to elements of a component must be combined to establish the overall component condition rating.

(3) **IFC standardization**: The interoperability of the MMBLMSs is of paramount importance because of the need to develop and use them by a large number of groups in a spatially and temporally distributed fashion. Standardization is important for facilitating data sharing and exchange between all the groups involved in bridge management at all the stages of the lifecycle. The Industry Foundation Classes (IFC) is an open international standard managed by the International Alliance of Interoperability (IAI) (IAI, 2004). In IFC2x2 the concept of visual presentation of geometric items has been added to the IFC model. Any object in IFC that has a geometric representation has two attributes: *ObjectPlacement* and *Representation*. The representation capabilities have two purposes: to add the explicit style information for the shape representation of products, and to add additional annotations to the product shape representations. ISO announced acceptance of IFC as a common language in the
construction industry in 2002. The IFC2x Platform Specification is now ISO/PAS 16739. The IFC-Bridge project aims to extend ISO/PAS 16739 by defining a standard representation for bridge life cycle management (IFC-BRIDGE, 2004). Examples of new entities defined in IFC-Bridge are IfcBridgeStructureType, IfcBridgeTechnologicalElementType, IfcBridgePrismaticElement and IfcBridgeBondingElementType. As IFC-Bridge is still in the early stage of development, many details are missing. For example, the truss type is not included in the definition of IfcBridgeStructureType. Several extensions of IFC are necessary to cover the later stages of the lifecycle of structures. Hassanain et al. (2000) proposed an IFC-based data model for integrated maintenance management. The proposed approach includes entities such as IfcCondition, IfcInspection, IfcRiskSchedule, IfcResource, IfcCostElement, and so on.

There are two resources related to time in the Resource layer of IFC: IfcDateTimeResource and IfcTimeSeriesResource. In IfcDateTimeResource, calendar date and local time are defined and functions about validity are also created. IfcTimeSeriesResource is new in IFC2x2. It defines two types of time points and related values: regular time and irregular time. In regular time series, data are updated predictably at predefined intervals. In irregular time series some or all time stamps do not follow a repetitive pattern, and unpredictable bursts of data may arrive at unspecified points in time. A typical usage of these entities is to handle data collected from sensors in a bridge health monitoring system.

(4) Requirements of space and time scales: MMBLMSs should link all the information about the lifecycle of a bridge to a 4D model of the bridge incorporating different scales of space and time in order to record events throughout the lifecycle with suitable LoDs. In the field of computer graphics, the basic idea of LoDs is to use
simpler versions of an object as they make less and less contribution to the rendered image. When the viewer is far from an object, a simplified model can be displayed to speed up the rendering. Due to the distance, the simplified version looks approximately the same as the more detailed version (Shamir and Pascucci, 2001). As for the time LoDs, different types of schedules have different time units, such as month, week, day, and hour.

(5) Databases requirements: A large project needs to store pertinent data for the lifecycle that can be used in every stage to help managers plan and organize their work efficiently. MMBLMSs should support distributed databases while providing the required security management for the access and update of the data (de la Garza and Howitt, 1998; Liu et al., 2002). Although relational database management systems are still the norm in BMS practice, object-oriented modeling and programming tools are widely used in software engineering and can greatly enhance the quality of the software because of their flexible data structure. A good combination of the two approaches is the object-relational approach for database development (McClure, 1997) which can relate the information in the relational database with the data structure of bridge components as described in object-oriented programs (Object, 2004).

(6) Mobile and location-based computing and user interface requirements: MMBLMSs should support mobile and location-based computing by providing a user interface that could be used on thin clients, such as PDAs and tablet PCs (Fujitsu, 2004), equipped with wireless communications and tracking devices, such as a GPS receiver. For example, in the case of a bridge inspector equipped with a mobile or wearable computer that has a tracking device, based on the location and orientation of the inspector and the task to be achieved, the system may display
information about the parts of interest within the focus of the inspector or navigation arrows to the locations where cracks are most likely to be found. The spatial database of the bridge and the surrounding environment, and the tracking devices attached to the inspector, make it possible to locate structural elements and detected problems and provide navigation guidance to these objects. In addition, all newly collected information can be tagged in space.

Tracking technologies can be grouped into four categories (Karimi and Hammad, 2004): (1) active source systems, (2) passive source systems, (3) dead reckoning systems, and (4) hybrid systems (Azuma, 1997). Active source systems require powered signal emitters and sensors specially placed and calibrated. The signal can be magnetic, optical, radio, ultrasonic, or from the GPS satellites. The main passive source systems are electronic compasses, sensing the earth magnetic field, and vision-based systems that depend on natural light. Electronic compasses are small, inexpensive, and accurate. However, like magnetic sensors, they have the problem of magnetic distortion when in proximity to metals. Vision-based systems use video sensors to track specially placed markers. Dead reckoning systems do not depend on any external signal source. For example, an inertial system measures the linear accelerations and rotation rates resulting from gravity using linear accelerometers and rate gyroscopes, respectively. Hybrid systems use multiple measurements obtained from different sensors to compensate for the shortcomings of each technology when used alone. One possible hybrid system is to measure position by differential GPS and inertial tracking, and orientation by a digital compass and tilt sensors. Differential GPS (DGPS) is based on correcting the effects of the pseudo-range errors caused by the ionosphere, troposphere, and satellite orbital and clock errors by placing a GPS receiver at a precisely known location (base station). The pseudo-range errors are considered common to all GPS receivers within some range. DGPS has a typical 3D accuracy of better than 3 m and an update rate of 0.1-1 Hz. Real-time kinematic GPS (RTK-GPS) receivers with
carrier-phase ambiguity resolution can achieve accuracies better than 3 cm (Kaplan, 1996).

(7) Decision-support requirements: Bridge management tasks are in general knowledge-intensive tasks demanding specialized study and practical training. A simple “help” functionality is not suitable for MMBLMSs because the users are in mobile situations and do not have the time to browse the documents provided by such functionality. Therefore, the knowledge necessary for each task should be knowledge-engineered in a way that it is readily accessible and applicable in a certain situation based on the task. Rule-based expert systems can be used to organize the knowledge pertaining to each group of tasks, e.g., inspection or maintenance, and these rules can be automatically activated in certain situations based on the context of the task that is executed using agents technology (Mizuno et al., 2002; Russell and Norvig, 2003).

(Insert Fig.1 and Fig.2 here)

4 FRAMEWORK FOR MMBLMSs

4.1 General Structure of the Framework

The general structure of the framework is shown in Figure 1. This structure is based on developing an object-relational data model, integrating a number of technologies and then using the data model and the integrated technologies to develop applications.

(1) Object-relational data model

The data model in the framework is an object-relational data model. Data are stored in a hierarchy from most
detailed elements, such as a deck panel, to the main bridge structures. Each object table is related to the sub- or super-tables. Apart from the bridge structures, activities occurring during the lifecycle are linked with related objects’ tables to add details about time, type of the activity, etc. The time entities in the database are defined based on the time resources definitions of IFC. In addition, definitions from IFC geometric model resources are used to create multi-representation of the 3D bridge model with different LoDs. The data stored in the database about the structure of the bridge are read automatically and a logical tree is created based on the structure. Figures 2 (a) and (b) show an example of an object tree representing a bridge structure and its table representation, respectively. The relationship between each group node and the element nodes branching from it is a part-of relationship. Through querying the database, the root of the tree is found and the root node is created. Then, queries are applied recursively to find other element nodes based on the data stored in the table.

(2) Technology integration

The core of the framework is a 4D model that integrates a spatio-temporal database covering the different phases of the lifecycle, and CAD 3D models of the bridges. Further integration is necessary with GIS, tracking technologies and multimedia information. A 3D map of the area covered by the MMBLMS is needed in the framework to permit the computations based on the location of the users. Using this map, the models of bridges can be based on geographic global coordinates. In order to create a 3D map, 2D layers can be draped on the Digital Elevation Model (DEM) of the same area. In addition, the location of the user can be tracked using DGPS and/or other tracking methods, and this location is used to navigate the user (e.g., to find the location of the next element to inspect) or to extract some information from the database (e.g., information about the inspection history of an
element at certain location) using the concepts of LBC explained in Section 3. The location of the user is reflected on the 2D map and the 4D browser. Multimedia information, including images and videos, can be captured and automatically added to the database using the concept of the Digital Hardhat.

(3) Applications

With the integrated 4D model, the framework can be used to develop many applications, such as visualization, analysis, and decision-making support. Visualization has powerful functions for interacting with the system in a virtual reality or augmented reality modes (Hammad et al., 2004). Users can query the database through the GUI or by picking a specific element, and get the results as visual feedback in the 4D model, e.g., information about the painting or rehabilitation history. Users can easily navigate in the 3D space using navigation tools (explained in detail in Section 4.2.1). Other important applications are spatio-temporal analysis applications, such as workspace analysis. The different workspaces for each activity can be generated and conflicts between these workspaces can be detected and resolved using a rule-based expert system approach (Guo, 2002).

Several design patterns are used in developing the user interface, such as MVC (Model-View-Control), observer, proxy, and façade patterns. The MVC design pattern is selected for the present framework (Potel, 1996). MVC is a widely used software design pattern that enforces the separation between the input, processing, and output of an application. Each of these components handles a discrete set of tasks, enabling loose coupling and the ability to change one component without affecting the others.

(Insert Fig. 3 here)
4.2 Computational Aspects of the Framework

4.2.1 Navigation Modes

Two types of navigation are suggested in this framework: logical navigation (Reinhardt et al., 2004) and graphical navigation. The logical navigation is represented by a hierarchical tree, which includes the structure of the bridge as extracted from the object-relational database.

The graphical navigation is achieved by interacting with the 3D model. Three navigation behaviors are investigated for the framework: drive, fly, and orbit behaviors. These behaviors use the pointing device (e.g., digital stylus or mouse) to control the view platform motion. Each button on the pointing device generates a different type of motion while the button is pressed. The distance of the cursor from the center of the coordinate system controls the speed of motion. As an example of the navigation behaviors, the drive behavior allows the user to move to any point in the 3D space, with pointer controls for translations along the X, Y, and Z axes and rotation around the Y axis as shown in Figure 3.

(Insert Fig. 4 and Fig. 5 here)

4.2.2 Picking Behavior

Interaction with the 3D model is mainly facilitated by picking the elements of the model. Picking is the process of selecting shapes in the 3D virtual world using the 2D coordinates of the picking device. In order to interactively retrieve or update information related to the picked element, it is important to know the location and the orientation of that element in the 3D environment of the virtual model. Figures 4 and 5 show the flowchart and an example of
the picking behavior, respectively. A pick shape is selected as the picking tool. The pick shape could be a ray, segment, cone, or cylinder. The pick shape extends from the viewer’s eye location, through the picking device location and into the virtual world. When a pick is requested, pickable shapes that intersect with the pick shape (e.g., pick ray) are computed. The pick returns a list of objects, from which the nearest object has to be found. After the closest object \((O)\) is found, the surface \((F)\) that faces the user should be identified to display suitable feedback. Through the calculation of the distance between the picking device position and the intersection points, the nearest intersection point \((P)\) can be found as well as the geometry of the face \((F)\) that contains \((P)\). The normal vector \((N)\) of surface \((F)\) can be calculated based on the current coordinates. The normal vector is used to represent the orientation of that face. Based on \(P\) and \(N\), the shape representing the feedback can be created and inserted in the scene graph at point \(P'\) with an offset distance from the surface \(F\) proportional to the size of the shape. The vector representing point \(P'\) can be found using the following equation:

\[
\hat{P}' = \hat{P} + \text{offset} \times \hat{N}
\]  

(1)

The following example of the visual feedback based on picking is given to illustrate the method of calculation (Figure 5). In the case of inspection, the system allows the user to directly add a defect, which is represented by a 3D shape, on the surface of the inspected element. The location of the defect is represented by the point \((P)\) of the picking. However, to show this defect on the surface, the center point of the 3D shape of that defect should be moved in the direction of the normal vector on that surface \((N)\) with a small offset distance based on to the size of 3D shape as shown in Figure 5. Otherwise, the defect on a thin element, e.g. the web of a steel beam, may appear on both surfaces of the web due to the small thickness of the web. The center point of the defect representation can be calculated using Eq. (1). Different defects can be represented with different shapes and the level of the defect
can be represented with different colors as shown in Figure 12 (to be explained in Section 5).

(Insert Fig. 6 and Fig. 7 here)

4.2.3 LoDs

The basic idea of LoDs is to use simpler versions of an object to meet different precision needs and improve the image rendering performance. When the viewer is far from the object, a simplified model can be used to speed up the rendering. Due to the distance, the simplified version looks approximately the same as the more detailed version. LoDs algorithms consist of three major parts: generation, selection, and switching. Generation is generating different representations of a model with different detail. Selection is choosing a LoDs model based on certain ranges for the distance. Switching is changing from one representation to another. When the user moves, this event is detected and the distance between the user and the object is calculated. Based on this distance, the corresponding switch will be selected and the model that should be displayed in this range is rendered (Figure 6).

Also, LoDs can be used in parallel with respect to different objects in the same system, such as the bridge element and the defects on the element. Each LoDs group uses different referential center point and distance range and operates only on objects related to that group. As shown in Figure 7, two LoDs groups can be used in parallel. LoDs Group-1 is for the whole bridge model, which includes five different cases: nothing shown, line, wire frame, prismatic elements, and detailed VRML objects. The distance $d_1$ is measured between the viewpoint and the center of the bridge. The distance range is defined in general depending on the bridge length. In this example, the visible range is from 0 to 20 times of the bridge length. LoDs Group-2 is for the defects on a floor beam, which includes
two cases: show or not show the defects. The distance $d_2$ is measured between the viewpoint and the center of the beam.

Although the above mentioned techniques (navigation, LoDs and picking behaviors) are common techniques in computer graphics, applying these techniques in MMBLMSs requires developing specialized methods that satisfy the special needs of these systems.

5 PROTOTYPE SYSTEM DEVELOPMENT AND CASE STUDY

To demonstrate the feasibility and usefulness of the proposed methodology, a prototype system is developed and is discussed in detail in this section. This prototype system is designed to fulfill the requirements discussed in Section 3 using the computational methods discussed in Section 4 to realize the following major functions: (1) Representing the 4D model of bridges with different LoDs; (2) Designing a user-friendly interface with access control that can be used in mobile situations; and (3) Developing comprehensive bridge databases including design, construction, inspection, and maintenance records.

(Insert Fig. 8, 9 here)

5.1 Case Study

Jacques Cartier Bridge is chosen as the subject of the case study. Jacques Cartier Bridge is a five-lane bridge with about 2.7 km in length, spanning the St. Lawrence River between the cities of Montreal and Longueuil (PJCCI, 2004). The bridge has a steel truss frame combined with prestressed concrete decking structure system. Inaugurated
in 1930, this bridge carries about 43 million vehicles per year with an annual increase rate of 2.4%, making it one of the busiest bridges in North America when considering traffic volumes per lane. Over the last 70 years, the old reinforced concrete bridge deck had suffered seriously from the increase of the number and load of trucks and the de-icing salts used extensively since the 1960s. Consequently, the deck was replaced in 2001 and 2002. This replacement project is the most significant restoration project ever undertaken on a Canadian bridge. During two construction seasons in 2001 and 2002, the bridge underwent complete re-decking of the five lanes. The new deck is constructed of precast, prestressed and post-tensioned panels made of high performance concrete which were prefabricated in a temporary plant installed near the south end of the bridge.

The bridge data were acquired from the bridge management authority (Jacques Cartier and Champlain Bridges Incorporated) (PJCCI, 2004; Zaki and Mailhot, 2003). The data include AutoCAD drawings, deck rehabilitation schedules and inspection and maintenance records. Figure 8 shows part of the inspection data of a floor-beam including metal loss and perforation. Figure 9 shows main span painting history of the bridge until 2003. These data have been used in the development of the prototype system. Several 3D models with different LoDs were created by converting the DWG file of the bridge into DXF (Data eXchange Format) and VRML (Virtual Reality Modeling Language) files and extracting the information about the geometry and topology of the bridge elements into our database. The database was built to include data about the different stages of design, construction, rehabilitation, and inspection. In addition, we acquired the digital map and the DEM data of Montreal to generate 2D and 3D maps (Clément, 2004). Furthermore, a number of simulations were developed to demonstrate the usefulness of the 4D approach, such as displaying elements with different colors according to construction, painting, or rehabilitation periods.
5.2 General Implementation Details

The structure of the prototype system follows the framework architecture explained in Section 4.1. The system integrates a 3D model of a bridge with an object-relational database, GIS and tracking components, and multimedia equipment to develop a 4D model for BMS that can be used on-site in mobile situations for retrieving and updating information. Using the 4D model, the user can directly interact with the system to get information on a certain stage of the lifecycle of a bridge. In order to allow for information sharing on the Internet, Java programming language is used to build the system. Java is a platform-independent and versatile language, enabling developers to create applets that can be downloaded and run within a web browser while interacting with server-side applications. Java 3D is used to implement the 3D graphics of the system (Walesh and Gehringer, 2001). Java 3D is a runtime API for developing portable applications and applets that can run on multiple platforms and multiple display environments. A digital video camera is connected to the system to facilitate image and video capturing. In addition, the system provides a rule-based expert system to support the decision-making related to inspection activities using Java Expert System Shell (JESS) (Friedman-Hill, 2003). The initial testing of the system was done using a Fujitsu LifeBook T4000 Tablet PC equipped with 1MB of RAM to improve the rendering performance. Because of the large scope of the system, the discussion in the rest of the paper will be limited to the main features of the system focusing on the overall architecture and user-interface design. Further details about the system can be found elsewhere (Hu and Hammad, 2005; Mozaffari et al. 2005; Zhang and Hammad, 2005).

The Graphical User Interface (GUI) of the system is developed using Java Swing classes. Because Java is a cross-platform language, the GUI components may have different sizes depending on the platform. Therefore, creating a proper layout manager is extremely important. A layout manager controls the size and position of
Components in a Container. Because the screen space of mobile computers is limited, tabbed panes are used to organize the different data items necessary at each stage of the lifecycle of bridges.

The 4D model is built using Java 3D based on the CAD drawings of the main span of Jacques Cartier Bridge and other data about the original construction and re-decking schedules. At this stage, only the bridge truss and the deck panels are considered. Virtual universes in Java-3D can be created from scene graphs. Scene graphs are assembled from objects to define geometry, location, orientation, and appearance of objects. Java 3D scene graphs are constructed from node objects using BranchGroups to form a tree structure based on parent-child relationships. TransformGroup objects can be constructed by applying Transform3D objects, which represent transformations of 3D geometry such as translations and rotations (Walesh and Gehinger, 2001).

5.3 Database Design

Java Database Connectivity (JDBC) is a programming framework for Java developers writing programs that access information stored in databases. The system has options to connect with several database management systems (DBMS) like Oracle, Informix, Microsoft Access, MySQL, etc. The commands to be executed by the DBMS on the database are based on SQL (Structured Query Language). In addition, in order to allow the system to interoperate with other applications, we use an IFC data structure representing bridge 3D objects (IFC-BRIDGE, 2004).

The database of the 4D model is designed with Microsoft Access to present the information of all the truss and deck components of the main span of the bridge. The name, type, dimensions, location, properties, and the starting and ending dates of the construction or maintenance activities of each member are defined in the corresponding
In order to avoid security restrictions resulting from the applet accessing the database directly, we use three-tier solution where the applet is only responsible for display, and will introduce mid-tier for all application logics that are related to data retrieving/updating, etc. The mid-tier can be realized using a servlet or a middleware application (CORBA or EJB) between the applet and the database. Another method to bridge between the front-end application and IFC or XML data is Web services where all data are saved in a central database. A web service application will provide services to query 3D bridge objects. The users can license the 3D model API to hook up their applications with the web services.

5.4 GIS and Tracking Components

A GIS sub-system is created using MapObjects Java Edition (ESRI, 2004). The purpose of adding the 2D map of Montreal is to provide information to the users of the system (e.g., bridge inspectors) about their locations and the environment around them. The map includes several layers related to Montreal City, such as a boundary layer and other layers for the roads, rivers, and administrative areas. The Modified Transverse Mercator Projection (MTM) was used because it is the standard projection used by the local government. The GIS has the main functions for zooming and retrieving information about the attributes of different layers. In addition, to locate the bridge model on the map, the same map of Montreal and the DEM were added to the 4D browser. The location of the inspector can be retrieved from the tracking devices and combined with the location of an element or a defect, which are registered in the spatial model, to help the inspector find his/her targets using virtual arrows.

Finding the location of the user is achieved using Differential GPS (DGPS), RTK-GPS, or video tracking. We
are testing the system with a Trimble 5700 RTK-GPS receiver. We are also using an Augmented Reality toolkit, called ARToolKit (Hirokazu, 2000), to track visual markers by means of a video camera. This method has many limitations on the accuracy of tracking and the range for recognizing a marker, which varies with the marker size. For example, a marker with an edge size of 20 cm can be recognized from a distance of about 150 cm. On the other hand, the GPS can be used only under the condition of having direct line of sight to at least four GPS satellites.

(Insert Fig. 10, 11, 12 here)

5.5 User Interface Design

5.5.1 General design

The main user interface of the system is shown in Figure 10. On the right-hand side, there is a time input interface that allows the user to query the database about events that happened during a specific period (e.g., Which parts of the bridge were constructed by the end of 1928? What is the sequence of replacing the deck panels in 2001?). The start and end dates of a period can be input using a calendar interface or sliding bars, and the 3D model will reflect the corresponding elements with different colors representing the progress ratio. A logical tree of the bridge structure is also shown on the right-hand side. Each tree node has a check box, which facilitates showing or not showing that element in the 3D model. In addition, the user can navigate the 3D bridge model and select an element of the bridge by picking that element. Upon selection, the element will be highlighted and the related information about the element will be displayed. Alternatively, the user can select an element from the database interface and the element will be highlighted in the model.
5.5.2 Inspection user interface

The inspection user interface is explained here as an example of the interaction methods used in the system at different stages of the lifecycle. An inspector can apply inspection procedures through a number of ordered tabbed panes. The panes are Inspector, Schedule, Element, Instrument, Damage, and Task. In the first two tabbed panes, some general inspection information needs to be input about the inspector and schedule. The user can find, add, and update the bridge inspection data by querying the database. In the Element pane, the inspector can choose the exact element to inspect according to a customized inspection scheme by picking the element on the 3D model at the approximate location of the defect. In the Instrument pane, a suitable inspection tool can be selected depending on the type of defect. Damage pane is the core part of the bridge inspection interface. Video/image capture functionality has been also implemented using Java Media Framework (JMF) API (JMF, 2004). The last pane, Task (Figure 11), is to summarize the previous inspection information for future assessment.

Figure 12 shows an example of picking a floor beam on the 3D model at different locations to input the location of defects. The defect will be automatically marked on the 3D model of the floor beam using a specific shape and color, which are defined based on the defect type and deterioration degree, respectively. For example, in Figure 12, the black sphere represents very serious metal loss.

Bridge inspection is a knowledge-intensive process. In order to support the inspectors using the system, the Java hyperlink functionality was added to allow the user to access inspection manuals in Hypertext Markup Language (HTML) format, such as “Bridge Inspector’s Reference Manual” (FHWA, 2002). The link is context sensitive and will extract only the relevant information. In addition, a rule-based expert system (Friedman-Hill, 2003) is developed to analyze the collected defect data and to calculate the element condition rating. The details of
the expert system development are beyond the scope of this paper and can be found in Hu and Hammad (2005). In the future, other functions for drawing sketches and generating history reports will be added to the system.

(Insert Fig. 13 here)

5.5.3 Levels of Details (LoDs)

Four different LoDs for the shape can be used in this system. Line, wire frame, prismatic elements, and detailed VRML objects are used according to the distance between the viewpoint and the model to optimize the performance of the system. As shown in Figure 13, when the viewpoint is far from the bridge, the user can see only one line representing the axis of the bridge. When the viewpoint comes nearer, the user can see the wire frame, prismatic elements and the detailed objects, sequentially. The concept of LoDs is also used to control the display of defects.

A calendar and sliding bar interfaces are used to specify a date or a period of time and the time step, representing the temporal LoDs, to be used in a simulation (Figure 10). Different temporal LoDs are needed during construction and maintenance periods. The year or the specific date of the maintenance action can represent the time of maintenance. For example, the painting of the main span was done in several years as shown in Figure 9. The inspection time is usually represented by the date of inspection. Higher time resolution is used for inspection purposes to record defects that could happen in a very short time.
6 SUMMARY AND CONCLUSIONS

This paper proposed a new type of Mobile Model-based Bridge Lifecycle Management Systems (MMBLMSs) and discussed the requirements for developing such systems. The proposed approach makes the first attempt to integrate 4D bridge models with BMSs and to make the resulting information accessible to mobile on-site workers with a suitable interaction model. The requirements and a framework of MMBLMSs were discussed including creating an object-relational data model, technology integration and applications development. Several computational issues for realizing the framework were also discussed, such as navigation, picking behavior and LODs. The developed prototype system integrates 3D graphics and a database to realize the 4D model of Jacques Cartier Bridge. The prototype system was demonstrated to engineers responsible of the bridge management and they gave positive evaluation. Furthermore, the preliminary testing of the system and its user interface showed that it has good potential for realizing future MMBLMSs because it was carefully designed and implemented to satisfy the specific requirements of these systems. Further development and testing of the system in practical situations are necessary to improve the functionalities and usability of the system. In addition, we are in the process of modifying the system for FM applications (Mozaffari et al., 2005).

ACKNOWLEDGMENTS

We would like to thank Basheer Khabir, Khaled El Ammari, Taotao Yong, XinSong He, Xiaoming Li, Tao Wang and Xiaohui Xiao from Concordia University and Yanlin Wang from Borland Software Corporation for their contribution in developing the prototype system. Also, the cooperation of Mr. Guy Mailhot from Jacques Cartier and Champlain Bridges Incorporated, Mr. Adel Zaki and Mr. Raymond Coté from SNC Lavalin in providing the data about Jacques Cartier Bridge and evaluating the prototype system, and Mr. André Clément from the GIS
Department of Montreal Municipality in providing the GIS data of Montreal are greatly appreciated.

REFERENCES


University. *Bauen mit Computern*, Bonn, Germany. VDI Verlag GmbH, Duesseldorf, Germany.


Fig. 1. General structure of the framework
Fig. 2. Example of the object tree (a), and its table representation (b)
Holding the left button down while moving the pointer up and down translates the view along the Z axis (zoom in/out)

Holding the left or right button down while moving the pointer left and right translates the view along the X axis

Holding the right button while moving the pointer up and down translates the view up and down

Holding the left and right button rotates the view around the Y axis

Fig.3. Drive navigation behaviors
Fig. 4. Flowchart of picking and adding defects
Fig. 5. Example of picking the 3D model for marking defects
(a) 3D sketch and (b) side view
Fig. 6. UML interaction diagram of LoDs behavior
LoDs Group 1:
Representations:
- LoD₁₀: $d_1 > 20L$
- LoD₁₁: $20L \geq d_1 > 10L$
- LoD₁₂: $10L \geq d_1 > 4L$
- LoD₁₃: $4L \geq d_1 > 2L$
- LoD₁₄: $2L \geq d_1 > 0$
- Detailed model
- Center of bridge

LoDs Group 2:
Representations:
- LoD₂₀: $d_2 > 2LB$
- LoD₂₁: $2LB \geq d_2 > 0$
- Defect not shown
- Defect shown
- Center of floor beam

$d_1$: Distance between center of bridge and viewpoint
$d_2$: Distance between center of beam and viewpoint
$L$: Length of bridge
$L_b$: Length of beam

Fig. 7. Relationship between distance and LoDs for the bridge and the defects on a floor beam
Fig.8. Example of floor-beam inspection information
Fig. 9. Bridge painting history
Pre-defined cameras
GIS interface
Navigation tree
Calendar for date input
Time slider
Color codes
4D browser
Pre-defined cameras

Fig. 10. Screen shot of the user interface of the prototype system
Fig. 11. Inspection task report tabbed pane
Fig. 12. Inputting the defect location by picking
Fig. 13. Different spatial LoDs of the bridge