Data Fusion Using Geographic Managed Objects

Erik Kjems
Aalborg University, Department of Civil Engineering
kjems@civil.aau.dk

The way we design our buildings and cities has not really changed a lot for decades. Drawing boards have been exchanged with relatively small 30" inch monitors, pens and rulers have been exchanged with advanced digital tools mostly though disturbing, making the creative process of design merely a frustrating one. So what have we gained from CAD. Certainly a lot, but mostly the possibility to combine and fuse projects. Simulating future use and behaviour, revealing design issues and failures before actually built. Still data fusion is a relatively new challenge albeit quite obvious trying to assemble models coming from different systems and vendors representing different professional domains. This paper discusses data exchange and data fusion in general and presents a new development, which gives the possibility to enhance data as intelligent objects opening a whole new paradigm for both data exchange and data fusion.

Keywords: *Data Fusion, CAD, Managed Object, Data Exchange, Virtual Machine*

INTRODUCTION

Exchanging data electronically has been a challenge since the dawn of the digital age. Evolving CAD systems then naturally were built with data formats suiting their own purposes fulfilling customers' demands and fitting the functionality and software capacity into one common data paradigm. These formats have evolved further into huge propriety strongholds, which are almost impossible to conquer when it comes to developing a parser that can read the data contained within the files, and understand every detail of it. The formats are being developed further continuously, making it difficult to accommodate the technological enhancements when data are supposed to be exchanged between the systems. Even within software companies one can observe the struggle to address the changes correctly from one version to the next, although one should expect at least company internal data exchange developments to work almost flawless. This problem can be experienced in many different kinds of systems not only within the area of CAD.

The only way out of the misery is to develop open standards (OS). Or is it? Standards do not suffer from drawbacks due to poorly documented proprietary file formats, but they do also suffer from errors due to poor implementations especially when formats grow into large complex object oriented hierarchical structures making the implementation a tough job. OS are a priori always behind the latest developments and technological innovations, since OS evolve due to implementations of developments and trends already there, only waiting to get implemented, agreed on, and approved by the organization behind the OS. Nevertheless, OS are probably the best way with the technology at hand to ensure a reasonable broad exchange of data between different CAD systems and also other systems, but also to fuse data into bigger and more complex data models.

What we face at the moment is the necessary not only to exchange and fuse data between different CAD systems, but also to combine data coming from different professional domains. Each domain uses specialized software within their specific area of design and construction. The only common denominator is a geographic reference within a building, construction site, city or whole country. In some countries for instance Norway the fusion of data coming from different domains has reached quite far even without the use of open standards, but with the use of primarily proprietary file formats like Autodesk's (DWG). We also see model servers, which can handle data input and output using a common platform and open exchange standards. OS certainly are on their way and for instance the IFC standard has reached quite far within the building community, and the Open Geospatial Consortium (OGC, [1]) is pushing forward undauntedly for open standards within a wide area of data exchange. One of the biggest challenges at the moment is the open area and its infrastructure, like roads, railroad and supply networks.

But the next wave of challenges in the construction data area has arrived already. The latest demand in connection with data exchange goes much further. We have reached the sensor age. "The Internet of Things" (Weiser 1991) and "Big Data" (Mashey 1997) is all over the media and the design and construction area is a vital part of that, simply because for instance buildings use an increasing amount of sensors for primarily maintaining purposes, but also safety purposes and not least ensuring the perfect energy consumption in a building. Cities use sensors for traffic control, intelligent lightning or citizen protection. The sensors can be placed virtually during the design phase, simulated and tested. The virtual representation coming from CAD systems can be used as user interface making it intuitive to handle sensors and information streaming from the sensors to the control devices. But how do we handle this kind of information coming from many different devices and sources and from many different professional domains? How do we exchange objects with complex real time related behaviour? Are open standards the only solution? The OGC is certainly dealing with sensors and doing an honest effort to predict the need of sensors in the future, but is it at all able to cover the developments in time?

At the Centre for 3D GeoInformation we have developed a completely new concept to handle 3D objects referenced to a dynamic geographic location representing (although not necessarily) an object of physical extend with for instance behaviour attached (Kjems et al. 2009)(Kjems & Kolar 2006). This concept implies many things ranging from the use of geocentric coordinates rather than map projected coordinates, allowing objects to change or to move geographically in a 3D information system environment, scaling the model space from nanometre to one light year and handling the aspect of time in a scalable way, but probably first of all executing the objects and allowing them to act out behaviour without any limits. The latter is possible through the use of a virtual machine. In our case we use Java Virtual Machine. We call this kind of objects Geographic Managed Objects (GMO). In the following the developed concept will be presented with focus on data fusion, understanding the challenge of exchanging data, collecting data from different domains into one coherent model, but also handling sensor information coming from different sources, being able to combine the data flow, and interpret information into meaningful messages.

THE CORE OF DATA EXCHANGE

As mentioned in the introduction, most of the data exchange carried out today is predominantly controlled by the vendors of the major CAD systems used within our business. Proprietary file formats have been used as de facto standards for a few decades and this will also be the case in the future. July 2008 Bentley and Autodesk, the two main players signed a historical agreement for the data exchange between their systems (Khemlani 2008, available at [2]). They agreed to make it easier to exchange data and improve the support for each others proprietary file formats. In a way this was an obvious step for the big vendors. Even though the CAD market involving 3D construction is insignificant compared to the revenue seen in other industries, one can chose to see this agreement as a way to oppose the efforts seen for instance within the buildingSMART alliance, where several major OS projects are conducted pointing at alternative possibilities using open data exchange.

One could go as far as raise objections to the way the big vendors present their BIM solutions, because they are mostly closed environments and a collection of lots of proprietary add-ons to their main CAD package, but this is probably an inevitable development seen from a commercial point of view. So the real BIM is now OpenBIM while BIM as general term is used mostly in connection with proprietary systems. Supporting OpenBIM opens the possibility to an open free data exchange following the standards proposed by the buildingSMART alliance among others. One could argue, why do we need open standards, when the big players are doing everything possible to deliver a turnkey system to their customers. There is at least one big argument. It must be in everyone's interest to have as many competitors on the market as possible. A building project is not only about the built environment. A lot of professional domains are involved in a project and there is a great amount of specialized software taking care of all the different parts through all phases of the building process. Large vendors buy this expertise on the market expanding their own software portfolio within their proprietary environment suffocating smaller rivals. Open standards give the possibility for small businesses to enter the market and provide perhaps a better solution for certain project issues. A real BIM (sorry OpenBIM) environment is allowing all kinds of professional software solutions to extract and provide enriched data from and to a data model representing a construction at any stage of the design, construction and maintenance process. Nothing less.

Unfortunately data exchange is still mostly done with the means of data files and appurtenant file formats instead of using logged secured access directly to a model database, which has been underway for quite some time and has been marketed as part of a solution without really being an open model-server.

As mentioned already, this paper wants to present a completely new way of data exchange, which can handle data fusion, data behaviour and different semantic standards in an elegant way, but before this can be done the following will explain how a traditional CAD software environment handles data, functions and data exchange. This is done because it will be easier to explain the new way of spatial data handling in a similar way.

In the following graphical presentation of a typical CAD software environment one can identify two CAD systems (application 1 and 2). Each system contains data and each system has built in functions, which can be used in connection with the data, thereby producing new data and enriching the data. Each system can run in any combination of operating-system and hardware as long as the CAD software has been developed for each combination as well. This unfortunately is not the case at all. If data need to be exchanged between the systems, files are generated using specific proprietary or non-proprietary formats, either or is not essential here. The same situation will occur if data are coming from a third-party provider, for instance application 3 or in figure 1 depicted as external data source. All functionality is part of the system (Function 1, 2, 3, 4,...). So if any functionality is missing in the application one can hope for the possibility to add it by installing available plug-ins, or aim for the next updated version of the application to include the particularly missing function. Within commercial BIM one can observe that sometimes whole new software packages are offered for sale to add functionality.

A somehow different, and yet not so common issue is, if data are supposed to be enhanced with behaviour,

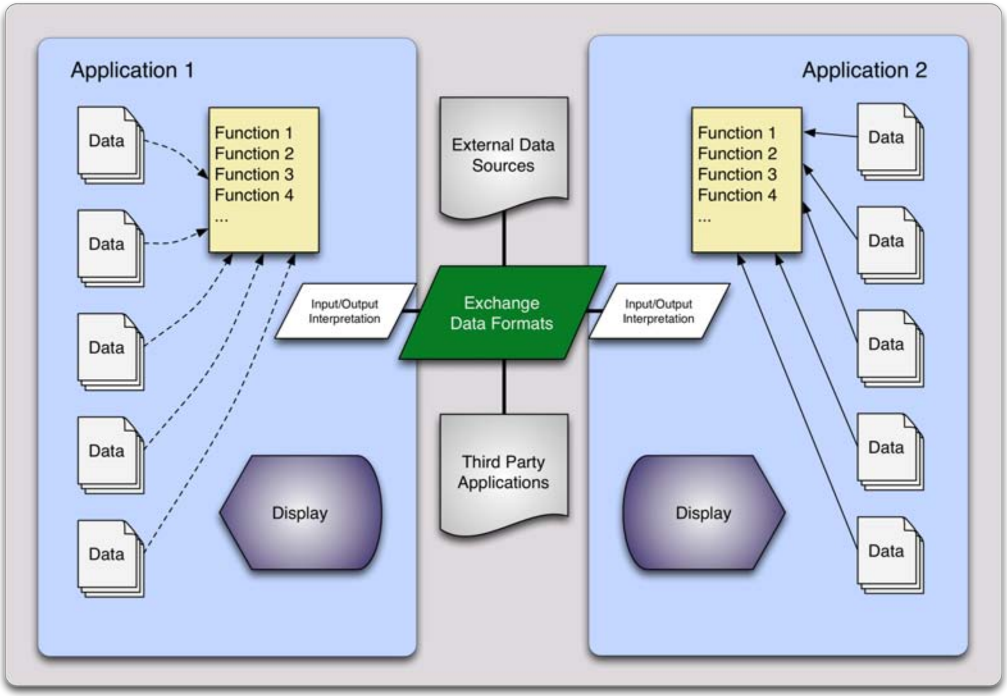

Figure 1
Traditional structure of applications and data exchange

for instance interactivity, the provided system functions need to be able to take care of that. This could be animated functionality in a model for visualization purposes, or far more sophisticated behaviour connected to the modelled objects enabling them to act on changes in the built environment. Adding communicative behaviour to objects turn them into agents where they can connect to their real counterparts in the physically built environment, or wherever one wants them to. With the growing amount and complexity of built in technology, increasing the possibilities of controlling and optimizing a building, not only with regards to energy consumption, but also user related demands to office climate and optimal maintenance, objects with advanced behaviour are not only a cool feature during design, but could actually be essential for monitoring the building and maintaining it. There is nothing more intuitive than a 3D model of a building representing it and all its components. Imagine all the movies where 3D models have been used to give the non-professional spectator an overview of a certain situation. A big or a few big monitors with an interactive 3D model is for most people far more intuitive to grasp compared to a dashboard with a diagram and hundreds of lights and instruments. While one can bring a laptop around the building the dashboard remains in the control room.

ENHANCING A MODEL USING GMO

The way we exchange data today using open standards is in principle divided into two parts. We create/use a schema like for instance XML which allows us to write and read data which fits into this schema easily, and further develop a schema for the required semantic content allowing us to interpret the information. This demands, that we have implemented a parser that not only can handle XML but also has a full implementation of the semantics in use. The drawbacks are obvious and most people have experienced the inadequacy exchanging data this way. (Faus & Wen 2012). Exchanging data using proprietary exchange formats are somehow similar although this process is not really transparent for a common user. Well the OS way may not be either, but at least there is an opportunity to.

A new concept developed originally by Jan Kolar at Alborg University is dealing with the data exchange in a different way using what we call Geographic Managed Objects (GMO) (Kolář 2006; Kolář 2013). To understand the GMO concept it is essential to understand an important prerequisite of the development. As presented above traditional software uses built in functionality to handle data input and output. A GMO can be created containing both geometry and behaviour. The new and perhaps more complex situation is shown in the following illustration.

The most essential part of the concept and difference to all other systems involving 3D objects is the use of a virtual machine. A concept actually described as early as 1966 to avoid for instance compatibility issues (Feldman 1966; Richards 1969). It could be a .NET solution but since .NET only supports Microsoft originated platforms the choice in our case fell on Java Virtual Machine, which runs on Windows platforms, Linux distributions and OSX on Macs. This is indicated in figure 2 by having a combination of operating system and hardware (1) and (2). In general the concept could have been developed in many ways but ended up with a prototype, which partly will be included in the following, showing the versatility of the concept. The prototype is an open source development, and can be found at [3].

Using a virtual machine first of all gives one the possibility to use data exchange in a binary form. The binary file format opens up a lot of possibilities, and is not a closed format as many think, as long as the underlying data format is open. The binary data format, which will be executed by the virtual machine when it enters the application, makes the whole difference. Imagine working on a model and receiving data from an external source. The received objects are automatically executed, which makes it possible for each of them to for instance "unfold" themselves into visible geometry since the functionality is part of the object. The object has been created with several classes perhaps hundreds, which will be used for instance to show its modelled geometry, to animate certain movements, or to allow to interact with the user because it also can bring along its own user interface, which can show up as a separate window. Clicking on the object will typically invoke the window. What the object can do or is supposed to do depends entirely on what is added as behaviour to the object. These kinds of objects are in figure 2 shown as boxes with both data and behaviour included. As minimum behaviour one should expect the object to appear as visual object. Perhaps one can imagine that the data exchange format now has moved to the core binary format of the Java programming language, which can be executed by the virtual machine. Java virtual machine is world wide the most used system for this kind of data exchange. Creating simple GMO from pure 3D data as they are used today in CAD software is not that difficult, and can be compared with an exchange of data using IFC with XML. One will in this case not need the semantic part though, because the objects reveal themselves executed by the virtual machine, and does not need to be parsed and recognised as certain building elements, they are self-explanatory. GMO are encapsulated in small byte-code files and can therefore easily be stored in databases, which even can be organised in a distributed way using a combination of time and space for the indexing.

In figure 2 one also can find boxes with only data or behaviour. GMO do not necessary have to carry

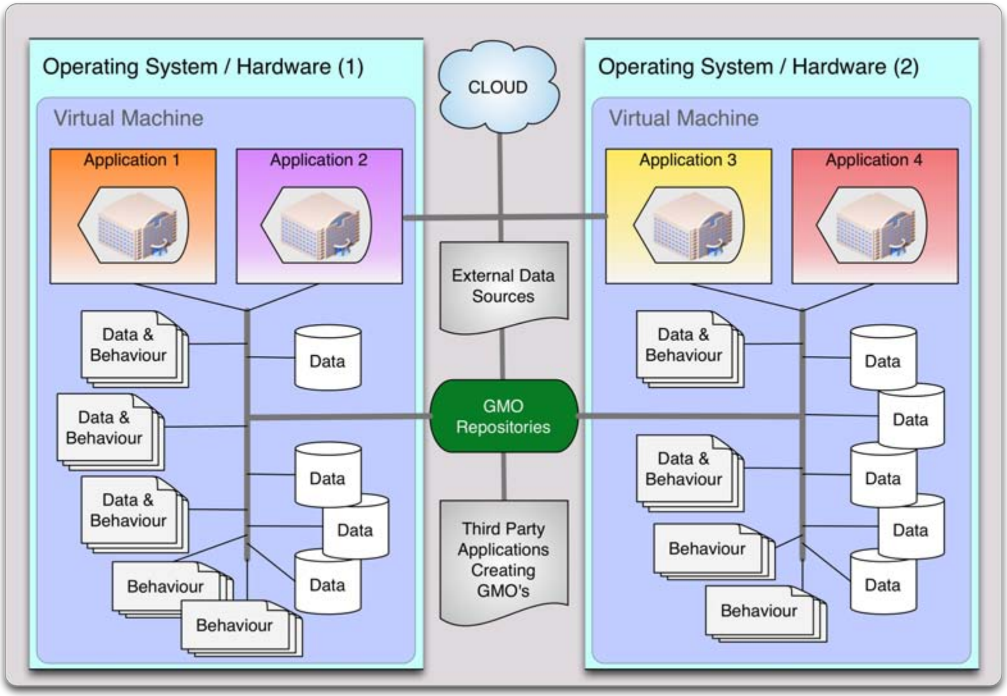

Figure 2
Virtual Machine and GMO based application and data exchange

everything with them. Supplementary data or behaviour can be fetched from other places internally or externally. If one for instance receives data from a vendor of building parts for instance certain window sections, these data represented by objects, can fetch the latest data update directly from the vendor referring to a production line or certain glass parameters. A GMO does not necessarily need a geographical extend, which makes it a MO. This way objects can represent an agent like functionality, and does not need to be visible at all, but handling functions in the model. This could for instance be clash detection, a distance rule, or detection on non-geometric rules controlling whether all rules are applied to by the building objects, and the overall design.

Just as an example it shall be mentioned, that for instance simulations developed for buildings will be handled the same way. Either by letting simulated people using the building, or letting wheel chairs moving around in the building, testing intelligent room lightning or what ever. If the objects are developed with a general approach, they simply can be replicated, and loaded into all kinds of models.

This might look strange and far away from what seems possible for most people because it seems as if programming is involved, but there is not more programming involved than in traditional systems. Lately Oracle seems to have recognized a similar approach for their solution for the Internet of Things [4] since they include among a lot of other parts Java Virtual Machine to handle sensors and its data with embedded Java software.

So GMO are not only containing the information about the object geometry and attribute data but can also contain functionality. Imagine a 3D model of a new site development with roads, buildings etc. where one wants to introduce lightning using different lampposts with a certain design and function coming from different vendors. With a bit of luck one can be able to get data in 3D, which could be implemented into the model with or without semantic information but probably with the right look. If one later wants to use the lamppost in a maintenance system one would need to extract the most important information for instance by adding it manually. If the lamppost was created as a GMO, first of all semantic rules could have been implemented. This means, the object itself can follow design rules. If one places a lamppost to close to another object it is not supposed to and thereby breaking a rule, it would react on it and for instance suggest moving to a different location close by, and thereby obeying the rule. Further the modelled lamppost "expects" to be connected to the real lamppost, when it has been put in place, and communicate with it directly from the representing data model, indicating for instance whether the bulb is turned on or off or not working properly at all, and perhaps even calling for maintenance. Every time this GMO is loaded into a model it will do the same. No semantic standard has been used, since the lamppost as a GMO carries all necessary information provided by the vendor during arrival to the model.

Exactly this situation more or less has been modelled as a case for a prototype that was developed in a research project conducted by a group of especially computer scientists coming from geo-science and engineering. The project had a volume of € 4M partly financed by the Norwegian Research Council. The cooperating parties were as follows: Iver from Spain, Vianova Systems Finland Oy, Norkart and Vianova Systems a.s. from Norway and Aalborg University, Denmark, where the latter added the basic development and fundamental ideas of the GMO concept into the project.

Figure 3 shows a situation where lampposts and manholes have been added to a city model. It must be stressed that the focus in the project was on the development of GMO, and the user interaction rather than beautiful textures and shading. The ground surface has been removed at the manholes to show the possibility of conflict between the objects.

The situation in figure 3 describes a situation where a lamppost and a manhole have been activated by clicking on it. This way they could be moved controlled in all directions (6 DOF on 2D monitor). The lower little window gives the possibility to activate the clash detection, which in this case would turn both objects orange indicating they are colliding with a distance rule, and suggesting new positions for probably both objects. Both objects are connected to other similar objects and many different rules could have been applied. If a third engineering or design domain would add objects to the model for instance traffic signs, they could be checked for any rule violence as well as suggesting an alternative position to either of the objects, until everyone is satisfied. This prototype allowed multiple users to alter the model simultaneously. In this case no user constraints were implemented, although it would be natural that each domain would be restricted to only have access to one owns objects.

This was a very simple example and others have been made. But the idea presented here should be very clear. Imagine all vendors of lampposts and manholes creating their own GMO providing them online during the design and construction phase. Each of the domain experts know exactly what kind of rules that do apply for the building components they produce, and could easily apply them to the virtual representation. Each time the model is loaded into the viewing application, the rules are applied in principle without anyone else knowing exactly what kind of rules are involved, although quite often they are common knowledge.

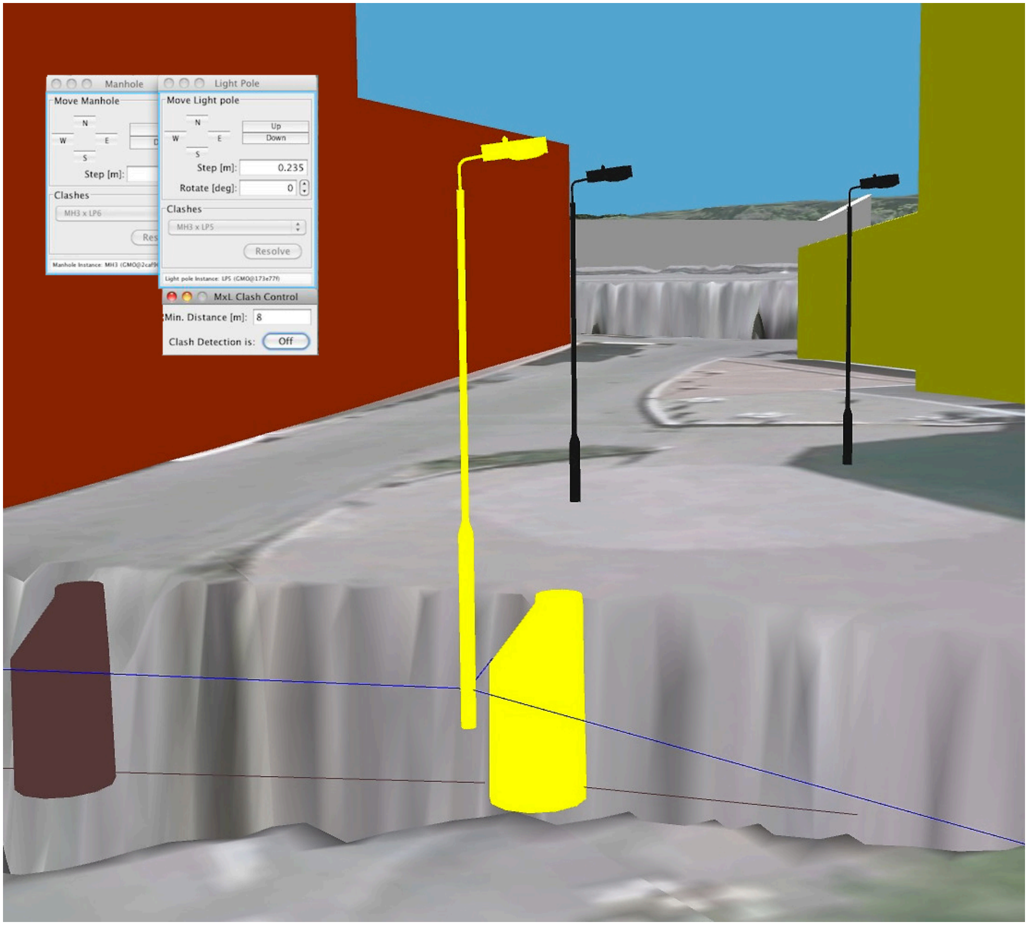

Figure 3
Example situation from the prototype with lampposts and manholes as GMOs

PERSPECTIVE

The development with the prototype showed clearly the huge potential of an open system where building objects are encapsulated structures, which unfolds themselves carrying all necessary information from geometry to design- and maintaining rules. As soon as they are loaded into the common design model they are activated, and objects with behaviour are able to react on from outside coming impacts, and to interact with other objects and users. As long as the GMO can be executed by the virtual machine one does not need anymore with regards to data exchange. Perfect for data fusion.

The drawback of this concept is that all existing CAD systems would need to be redesigned from scratch. It would though be possible to create an application like Solibri Model Checker that could handle GMO. Most GMO especially in the first period of time would mainly contain geometry perhaps one GMO will simply take care of objects containing geometry altogether. It would be quite easy to create software that could take IFC objects and making them into GMO. Enhancing them with behaviour could be made as drag and drop options within each domain.

Most of the challenges have been discussed and sorted out conceptually during the research project. But this is too big an assignment to carry out for small development groups. Big companies should have the resources to try it out and put it to market. Firstly as a supplemental to traditional packages later as a fully implemented concept of distributed model servers with hundreds of applications creating GMO.

Millions of Euros or whatever currency is preferred are lost every year due to design and construction errors, which have to be fixed on-site (Lopez & Love 2011). Fuse all available data during the design phase and let the model check for any kinds of conflicts both geometric clashes and semantic clashes. The GMO concept is a way to handle that. Looking only a few years ahead with sensors everywhere, a 3D interface would provide an intuitive visual platform, and give connectivity to a variety of sensors on the front, while collecting data interdisciplinary on the back site, connecting them with smart algorithms to provide a bigger picture, and better decision support. The latter is actually one of the biggest challenges to face regarding sensor data. The GMO concept can provide the platform for that. There are so many advantages of the GMO concept that it should be obvious to start developing in this direction. As mentioned, Oracle has been starting off, and that could be a turning point, but whether the big players within CAD and GIS will turn their ships in another direction or perhaps put a new boat into the water, well, that is somewhat a different story.

REFERENCES

Faus, JD and Wen, W 2012, 'Improving Interoperability of 3D Geographic Features via Geographic Managed Objects', *International Journal of 3-D Information Modeling*, 1(1), pp. 43-54

Feldman, JA 1966, 'A Formal Semantics for Computer Languages and Its Application in a Compiler-compiler', *Communications of the ACM*, 9(1), pp. 3-9

Khemlani, L 2008, 'Autodesk and Bentley', *AECbytes*, AECbytes "Building the Future" Article(July 10, 2008)

Kjems, E, Bodum, L and Kolar, J 2009, 'Managed Objects for Infrastructure Data', in Lee, J and Zlatanova, S (eds) 2009, *3D Geo-Information Sciences*, Springer Verlag, pp. 97-107

Kjems, E and Kolar, J 2006, 'Spatial Object Structure for Handling 3D Geodata in GRIFINOR', in Abdul-Rahman, A, Zlatanova, S and Coors, V (eds) 2006, *Innovations in 3D Geo Information Systems*, Springer, pp. 107-118

Kjems, E and Kolář, J 2006, 'Spatial Object Structure for Handling 3D Geodata in GRIFINOR', in Abdul-Rahman, A, Zlatanova, S and Coors, V (eds) 2006, *Innovations in 3D Geo Information Systems*, Springer, pp. 107-118

Kolář, J 2013 'Bytecode Unification of Geospatial Computable Models', *FOSS4G 2013 Proceedings*, Nottingham, pp. 26:1-26:19

Kolář, J 2006, 'On The Road To 3D Geographic

Systems: Important Aspects of Global Model-Mapping Technology', in Abdul-Rahman, A, Zlatanova, S and Coors, V (eds) 2006, *Innovations in 3D Geo Information Systems*, Springer, pp. 207-223
Lopez, R and Love, P 2011, 'Design Error Costs in Construction Projects', *Journal of Construction Engineering and Management*, 138(5), pp. 585-593
Mashey, JR 1997 'Big Data and the Next Wave of InfraStress', *omputer Science Division Seminar, University of California*, Berkeley
Richards, M 1969 'BCPL: a tool for compiler writing and system programming', *Proceedings of the May 14-16, 1969, spring joint computer conference. AFIPS '69 (Spring)*, New York, pp. 557-566
Weiser, M 1991, 'The computer for the 21st century', *Scientific american*, 265(3), pp. 94-104
[1] http://www.opengeospatial.org
[2] http://www.aecbytes.com/buildingthefuture/2008/AutodeskBentleyAgreement.html
[3] http://griffnor.net
[4] http://www.oracle.com/us/solutions/internetofthings