

# Development of Simulation System for Lift Design

A. Godwin\*, A. Ordys\*\*, J. Francik\*\*, P. Smolenski\*, J. Beebe, K. Zienowicz\*\*, R. Burley\*, Chan Meng\*, Ai Li\*

\**Lerch Bates Ltd., Woking, Surrey, UK (e-mail: adrian.godwin@eu.lerchbates.com)*

\*\**Kingston University, London, UK (e-mail: a.ordys@kingston.ac.uk)*

---

**Abstract:** This paper addresses the work done in pursuit of an expert system for designing building passenger vertical transportation systems. The design and specification of lift systems for large buildings is a very complicated process, with a wide range of variables that need to be evaluated in order to design a system that will deliver acceptable performance. Such evaluation is performed with the help of computer simulation software. The limited functionality of the currently existing programs for lift simulation however prompted the authors to pursue the development of a new system, suitable for the demands posed by modern building designs and improved control algorithms. The authors have completed the calculation and simulation modules together with the generation of the visual simulation and building information model and the approach and methodology employed is described within this paper.

*Keywords:* dynamic simulation, control of elevators, software systems, visualisation software.

---

## 1. INTRODUCTION

The design and specification of lift systems for large buildings is a complex process. This complexity is demonstrated by the fact that many systems already adopted in high rise buildings are considered sub-optimal. Large modern buildings often incorporate shuttle lifts that take people to a sky lobby, from which local lifts take people on to their final destination and, to date, it has not been possible to model this whole system.

In the late 1970's the use of computer simulation to evaluate lifts systems was introduced by Barney and Dos Santos (Barney and Dos Santos, 1977).

Around 1998 a PC-based simulation program was developed for general use by Peters (Peters, 1998)

This program, called ELEVATE, enabled users to model most types of buildings with associated lift control systems and types of traffic. It could not look at multiple groups of lifts in operation in parallel nor did it account for designs involving sky lobbies, double deck lifts with new so-called "destination" hall call control or address the various other important parameters of design including building space taken, capital cost estimates or the generation of a 3-D Building Information Model of the proposed lift services.

Currently the "state of the art" from the viewpoint of independent lift system simulation resources is the availability of either "PC-LSD", a program developed by Barney or "Elevate" a program developed by Peters.

The limited functionality of the current programs means that the designer must have considerable experience and tacit knowledge of how to address all the variables involved. The current systems are also not compatible with the planning requirements of modern lifts which now increasingly incor-

porate destination hall-call control systems where users "book" their calls even before they enter the lift lobby.

Recent advances in software development have provided the opportunity to develop an Expert System based upon a Building Traffic Simulator. Such a system would allow consultant's expertise in the form of rules that could be captured together with their in-depth knowledge of lift system design, to meet a wide range of needs, including:

- 3D calculation of the space taken, i.e. physical dimensions of a system that can be transferred as a Building Information Model into architectural CAD packages. This enables the building efficiency to be calculated i.e. the net to gross rentable space figure.
- Graphical performance output demonstrating the system's performance under differing traffic situations, efficiency curves and information in graphical format.
- Visual demonstration of performance to interested parties e.g. architects and developers enabling any long wait passengers or substantial queuing to be seen in a visual simulation of the operation of the lift services.

## 2. SIMULATION STRUCTURE

Computer simulation is especially valuable for large buildings when the traffic patterns are more complicated. Primarily simulation allows the relative capacity of different elevator configurations to be analysed, especially for peak traffic, in terms of "quality" and "quantity" of lift service.

The traffic simulation solution developed uses the concept of a so-called 'simulation platform' which performs all the underlying functions and processes of car movements and passenger activities. As seen in Fig. 1, the simulation platform consists of four parts; passenger generation, car jump model, individual car controller and the group controller. The group

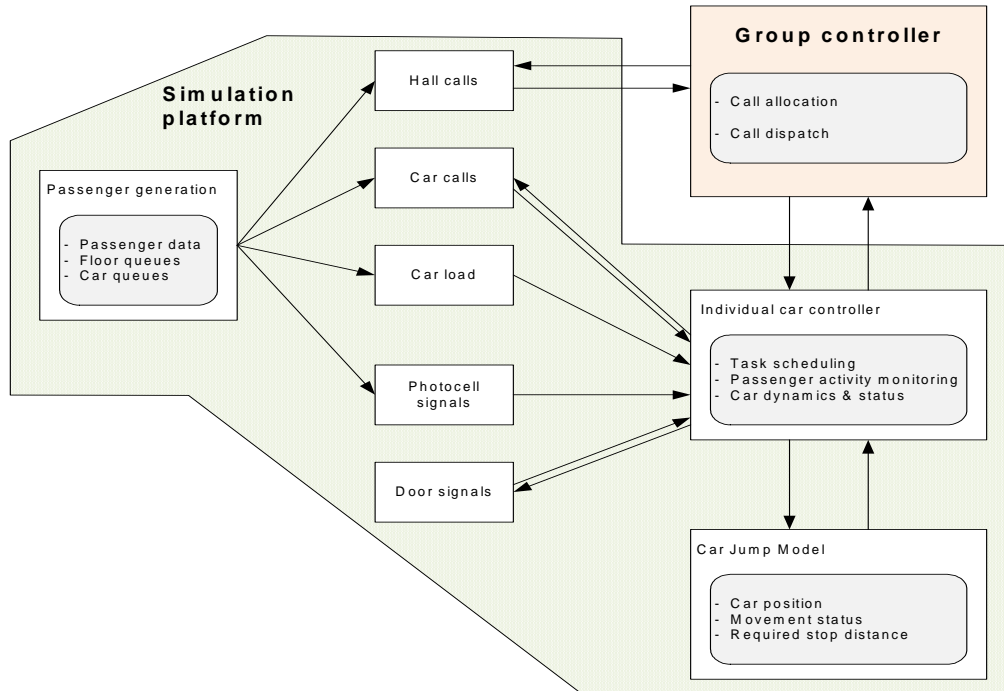


Figure 1: Elevator traffic processes (adapted from Siikonen 1997)

controller implements the passenger allocation to individual lift cars and is responsible for overall performance optimization. It will be described in more detail in the next section. Firstly we discuss the other three elements.

**Passenger generation** is the starting point for traffic simulation. The arrival and destination floors of individual passengers are random but they statistically follow a given or measured population distribution in the building through the destination probability matrix,  $d_{ij}$  which reflects the spatial geometry of the building or the relative attraction of particular destinations.

The Poisson arrival process takes account of passenger arrival time. Passenger arrival floor and destination floor are considered as a spatial distribution which is determined by the traffic pattern and the relative attraction of individual floors. Passengers are externally generated and are introduced to the model one by one at their arrival time. Every passenger is listed in the queue information and given a unique identity number. The queue status (ON /OFF) of the floor is triggered by the presence of passengers waiting at that landing.

**The car jump** model describes the movement of the elevator car within the shaft. It is commonly assumed that the car can attain instantaneous values of the rate of change of acceleration, which is termed jerk. This motion is described in terms of position ( $s$ ), speed ( $v$ ), acceleration ( $a$ ) and jerk ( $j$ ). Therefore, the car movement in a jump is given by:

$$s = \int_0^t v dt, \quad v = \int_0^t a dt, \quad a = \int_0^t j dt \quad (1)$$

where,  $j = \dot{a}$ , and  $c$  is a constant depending on the specification parameters of the elevator. Other important constants are maximum acceleration and maximum velocity.

There are basically three jump scenarios in typical floor to floor car movements:

1. The lift reaches full speed and full acceleration. This is typical for a low speed lift moving from floor to floor or a high speed lift making a multiple floor run.
2. The lift reaches full acceleration, but not full speed. It is typical for a short distance run for a high speed lift, where the distance travelled is not enough to accelerate up to and decelerate down from full speed.
3. The lift does not reach full speed or acceleration. This scenario corresponds to very short trips, e.g. single floor.

The **individual car controller** coordinates the operations of each individual car such as loading and unloading passengers selectively upon the car stopping and control of next destination stop for car movement. It takes account of functions inside the car, e.g., registering and cancelling of car calls, opening and closing of the door, passengers loading and unloading and measurement of the car load. It has two major functions, one for jump task scheduling, which means taking the hall call allocated from the group controller and car calls registered by in-car passengers to issue a destination signal for the car jump model; the other is monitoring passenger activity, which includes selective passenger loading and unloading upon car stops, appropriate passenger identifying methods compatible with passenger tracking and logging their every movement in the activity log.

### 3. THE GROUP CONTROLLER ALGORITHM

The algorithm represents the software installed in the group controller, which decides when, how and what to communicate to the car controller. The group controller selects one of the algorithms to be used to allocate passenger (hall) calls to an appropriate lift (car) which can service that call.

The algorithm is event-driven and is executed at least each time a new call is registered. It is also executed to re-assess any existing allocations of calls (if re-allocation is allowed) only when a significant event occurs, such as: *hall call registered, car call registered, door closed, call cancelled etc.*

For each lift in the group, the Algorithm uses the current state of the lift and all its registered calls to create a journey plan for the lift as it follows a Simplex Collective algorithm (*answering all its registered calls at floors encountered in its current committed direction of travel to one end of the shaft, then reversing and answering all its registered calls in the opposite direction, then finally answering all its registered calls for its current committed direction of travel that are at floors currently behind the lift*). The journey plans include relative timings of arrival at each planned floor stop, which represent the estimated time to cancel the registered calls at the floor for the current committed direction of travel.

The algorithm then makes a trial allocation of a new call by adding it to the travel plan of each lift in turn. It calculates the cost of allocation as the difference between the measured cost parameter before and after the trial allocation. The cost function calculation may be based on any calculated value: distance to travel to the call floor, average call waiting, average

journey time, etc. The algorithm stores the returned cost of allocation against the relevant lift. Finally the algorithm selects the minimum cost allocation and stores the allocation permanently against the hall call.

Two types of hall call are considered: **Conventional** (Directional) calls and **Destination** calls.

Calls are *Conventional* when only the Origin Floor and the requested Direction of Travel is known. In general, only one conventional call can be registered per floor per direction at any one time.

Calls are *Destination* calls when both the Origin and the Destination Floors are known and the requested Direction of Travel may be computed by comparing them.

### 4. SOFTWARE DEVELOPMENT STRUCTURE

The application is developed as four distinct sub-systems (Fig. 2): *Calculation, Simulation, Visualisation* and *Reports*, which can be selected independently and which are linked together via a user front end. Extensive use of XML as a means of passing information around the application allows the system to be customised for the applications usage and for fast transfer of data. No database is utilised removing the accompanying drag on resources that is needed to run a database engine. The coding language C#(C Sharp) is used for the majority of the application. This choice of language offers decreased development times compared to using C++ for the small loss in performance (estimated at 3%). It is also the language of the .NET environment, which allows the importing of other DLL's written in languages such as C++ and COM to be accessed via C# code.

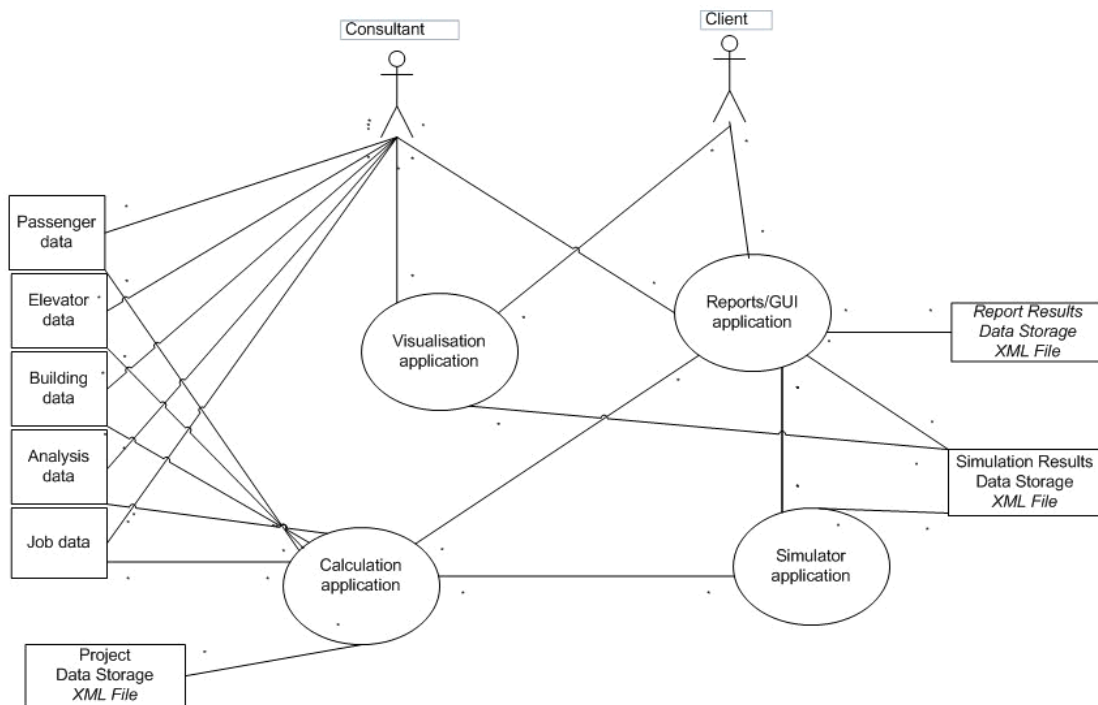


Figure 2: Software architecture

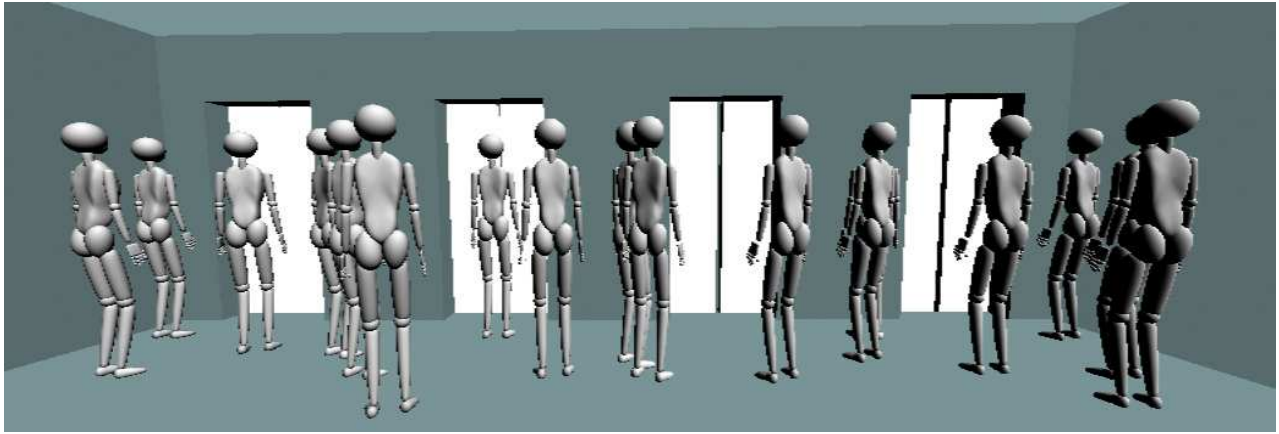


Figure 3: Lift lobby rendering

This helps future proof the application as add-ons can be written in the developer's language of choice. Linq is utilized for querying data held in the XML files. Such data includes known values for lift speeds and capacities as well as the accompanying sizes for lift cars, lift shafts, machine room data and the size and shape of the building's lobby.

### 5. 3D DRAWING INFORMATION AND VISUALISATION

There is a growing movement in the architectural world to produce drawings in 3D. One of the main advocates is Autodesk with their product Revit3D. Our application constructs a Building Information Model (BIM) for the essential 3D structure of the lift shafts and lobbies and exports it applying IFC – the Industry Foundation Class (IfcWiki, 2009), a file format that is now a well established standard, based on the latest International Alliance for Interoperability (IAI) IFC 2x3 data exchange standard, compliant with ISO-10303-21. This model is built using either data directly involved in the simulation process or automatically derived from the specification of the lift system chosen by the user, with most parameters exposed for further manual tuning. Once the final structure of the lifts, shafts and lobbies is decided, the information model may be easily and effortlessly fed forward to the architects using CAD tools.

The 3D structural model of the building is also used internally, by the 3D Visualisation Subsystem. This module, applying heavily the 3D animation techniques and based on FreeWill+ animation framework (Szarowicz et al., 2005) provides a facility to observe the movements of passengers and elevator cars in the building. Although not an indispensable element of the simulation tool, it may strongly influence the decision making process as it provides highly visual content to illustrate the performance of the lift services. For example, by introducing colour coding of the intending passengers based upon their expected waiting times, an instant visual cue is provided that allows the user to easily spot queues of lift users with excessive waiting times.

The visualisation module renders the building structure and populates it with 3D animated human characters (Fig. 3). They are driven by the simulation subsystem outcome, treated as a specification of their behaviour – or as a “script”. This is a major challenge as the information thus supplied is rough and contains mainly arrival times at various points of the space. To achieve the proper ‘look and feel’ a normal

human-like behaviour must be reconstructed, or simulated, automatically by the system. The population of passengers is therefore modelled as a swarm of autonomous distributed agents. The goal of each individual is to fulfil their “script” but, in the same time, to behave like a human, and first of all to avoid collisions with other passengers as well as architectural elements. To achieve this classical swarm intelligence approach [Reynolds, 1987] has been applied.

### 6. CONCLUSION

The new design tool developed will enable a more informed selection of lift design to be made by comparing the relative traffic performance, space-take and capital cost of each solution.

It will also be possible to quickly analyse the effect of changing input variables e.g. occupational densities upon the performance of the lift system.

Future plans include further development of the system such that the interaction between the visualisation and the simulation modules will be able to be performed on-line. In addition this new tool will enable the designer to ultimately reconcile the performance of a theoretical lift system with a “real world” system whereby the impact of the variable loading time for a car due to the position of passengers in the lobby etc. can be explored.

### REFERENCES

- Barney, G. C. (2003). *Elevator traffic handbook: theory and practice*. Spon Press.
- CIBSE Guide D (2005). *Transportation systems in buildings*. CIBSE.
- IfcWiki (2009). Industry Foundation Classes wiki website, <http://www.ifcwiki.org>, accessed 10/07/2009.
- Reynolds C. (1987) Flocks, herds and schools: a distributed behavioural model, *SIGGRAPH 87*, 25-34.
- Siikonen, M. L (2007). *Planning and control models for elevators in high-rise buildings*. Ph.D dissertation, 1997.
- Szarowicz A., Francik J., Mittmann M., Remagnino P. (2005). Layering and heterogeneity as design principles for animated embedded agents. *International Journal of Information Sciences*, 171(4) Elsevier, pp. 355-376.