“CLOCWiSe”: Constraint Logic for Operational Control of Water Systems

Mietek Brdys, Tom Creemers, Hans Goossens,
Jordi Riera, Anton Heinsbroek, Zbigniew Lisiak

1. Introduction

Management and operational control of water systems have to meet many requirements from different fields, in order to satisfy consumers with sufficient amount water of appropriate quality, in a robust, reliable, and cost-effective way. Fulfilling this task requires solving on-line complex, hybrid, combinatorial problems, dealing with inherent uncertainty of demands and quality, and asks for fast responses to unexpected emergencies. Traditional approaches of decision support can hardly cope with this complexity, because of the highly interactive nature of the problem that necessitates an integrated approach. Constraint Logic Programming (CLP), an innovative programming paradigm, appeared very fruitful in a project, called CLOCWiSe, by a consortium consisting of the authors’ institutions, and financially supported by the European Commission. It allowed building a Decision Support System (DSS) for the drinking-water world offering functionalities not present in existing applications in the field. CLOCWiSe breaks with a simulation-based approach which generally leaves the combinatorial aspects of the problem to the user and has difficulties to deal with uncertain and imprecise information. CLP offers a highly expressive programming tool which efficiently solves combinatorial problems. The approach is to reduce the initially large domains of possible values for variables by making these domains consistent with the constraints that implement the properties of the problem elements. This reduction approach not only offers a very natural way to express human knowledge but is also very suitable for dealing with uncertainties and incomplete information. Expressing the management and control objectives and the problem properties in the form of constraints allows for fundamental mathematical integration of different aspects of the problem. CLP can then be used to derive an efficient problem solver. In this paper a basic description of CLP is given as well as the concepts used in CLOCWiSe.

2. Functionalities and Structure

There are three main functionalities in the CLOCWiSe prototype:

- *network configuration management and control*
- *estimation of model parameters and network states*
- *emergency handling*

There is no dedicated functionality for demand prediction and the demand curves must be provided by an external facility. An operation of the functionalities’ software modules is illustrated briefly in Figure 1.

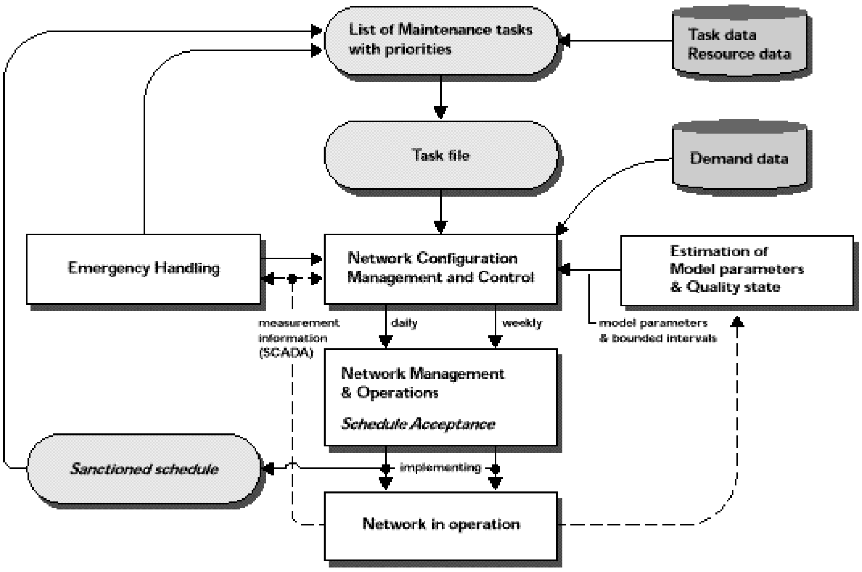
List of Maintenance tasks with priorities
Task data
Resource data
Task file
Demand data
Emergency Handling
Network Configuration Management and Control
Estimation of Model parameters & Quality state
measurement information (SCADA)
daily
weekly
model parameters & bounded intervals
Network Management & Operations
Schedule Acceptance
Sanctioned schedule
implementing
Network in operation

Figure 1. Software Architecture.

The first functionality handles in an integrated manner the network *maintenance scheduling* and the *network configuration* for integrated operational control of quantity and quality purposes. It generates on-line, within a repetitive operational decision and control structure, optimised maintenance schedules and pump and valve schedules. So far the implementation covers only networks with weak hydraulic couplings [Brdys 94] and treatment of quality is limited to mixing and transportation mechanisms. The implementation covering strong hydraulic couplings and reactions of the quality constituents is being carried out based on the most recent release of the CLP language, Chip V5 (See paragraph 3.1).

The one-week time horizon needed for the maintenance scheduling compared with the 24-hours time horizon and one-hour time step required by the network reconfiguration scheduling can lead to not tractable dimension of the integrated problem. Time aggregation is used to derive hierarchical two-time scale solvers capable of solving the integrated problem for large networks. Using the aggregated time scale requires aggregation of the constraints. The aggregation accuracy needs to be controlled to guarantee that feasible schedules can be obtained. The accuracy needs also to be considered from the view point of achieving a sound compromise between problem dimension and loss of optimality of the schedules produced. The constraint consistency reports are then produced on-line, based on information currently available and appropriate actions are undertaken accordingly. As the network model and prediction of uncertain inflows and demands are used in calculating the schedules, the model responses are systematically compared with the network measurements. The model-reality consistency is evaluated and the solution is recalculated, if necessary.

The consistent maintenance schedules are sent to Network Management and Operations Unit for acceptance. If accepted, their implementation is done by the NMOU staff. There are two possible mechanisms for the *emergency handling* functionality: *what-if* dialogue and producing a new maintenance schedule by means of the first functionality. At present only the latter mechanism is implemented.

The *estimation functionality* delivers on-line the constraint model parameters and estimates both the measured and not measured network variables. The variable and parameters are estimated jointly using model and measurement information available. Robustness of the estimates is achieved by employing a set-bounded model of uncertainty in the modelling and measurement errors. Stability of the interval-estimates is provided by a moving information window [Brdys 99] and their tightness can be traded off against computational efficiency by controlling the window length. The details concerned with this functionality design can be found in the accompanying paper [Brdys 99a]. At present the estimation is limited to the mixing quality and the hydraulic information is assumed to be available on-line.

3. Technology

3.1 Constraint Programming

Constraint Programming (CP) is a relatively young programming paradigm, world-wide subject of active research, which has shown, through numerous industrial applications, its potential to tackle difficult combinatorial problems, such as job scheduling in manufacturing, resource allocation, placement problems, routing, configuration, production planning and many more. A survey of practical applications can be found in [Wallace 96].

This *declarative* paradigm results particularly elegant and attractive when *constraint solving* is combined with Logic Programming (LP) into *Constraint Logic Programming* (CLP). LP is a tool which separates the *specification* part of a problem (i.e. *what* is to be solved) from the *procedural* part (i.e. *how* to do it). In this way, the programmer can concentrate on the constraints that have to be satisfied in his/her problem, while leaving the tedious task of programming the search through the combinatorial space of possibilities to the built-in LP engine. Additionally, the underlying constraint solving techniques will guarantee that the choices made will be consistent with all constraints and that unfeasible combinations will be pruned away early. The latter highly improves the efficiency of the resulting CLP language, while the possibility to express constraints from various domains (a natural way to represent human knowledge) provides its increased expressiveness.

CLP languages differ in the way constraint solving techniques are embedded in a LP framework. The approach based on *local propagation* over variables with *finite domains* [Dincbas 88, Van Hentenryck 89] showed one of the most powerful paradigms and rapidly penetrated industrial application domains. CHIP (for *"Constraint Handling in Prolog"*) was the first and most innovative prototype implementing this technology. It was initially developed in an industrial laboratory, the ECRC (European Computer-Industry Research Center) based in Munich (Germany). This CHIP project, finished in 1990, triggered much of what has followed in the CLP field since then. The CLOCWiSe software was entirely written in Chip v5[1], the latest commercial version of the language.

3.2 PLANETS - "Planning Activities on Networks"

The origin of the CLOCWiSe project lies in the work on PLANETS, a constraint-based scheduling system for power-distribution networks. A more detailed, technical description of PLANETS and of the way CLP was used, can be found in [Creemers 95, 98]. PLANETS, developed at the UPC and financed by the Spanish electricity utility ENHER S.A., was the first system to use CLP techniques in power distribution and also the first system ever able to solve *simultaneously* the complex *maintenance-scheduling* problem and associated *network-reconfiguration* problems, yielding *global solutions* to the overall problem, something that was very difficult or impossible with the traditional Operations Research, Mathematical Programming or Artificial-Intelligence techniques.

The success of the PLANETS project can be ascribed to the integration of some of its major problem aspects: *scheduling*, *reconfiguration management*, *load balancing* and *resource management*. Any of these aspects is generally seen as a typical case for a constraint-based approach.

By active reasoning on the problem's constraints (electric, topological or temporal), the CLP approach handles the entire problem in all of its aspects. Indeed, where traditional problem-solving techniques decompose a problem into sub-problems, in CLP it makes often sense to merge two interacting problems into one to make solving both of them easier. Constraints from one problem can propagate information to the other one and hence initiate additional pruning of its search space. In PLANETS this is the case with a *network-reconfiguration* problem and a *scheduling* problem. An additional advantage of this approach is that the final solutions will be *global* ones, taking into account all problem constraints at the same time.

3.3 Constraint model of a water-supply management problem

The similarities with water-distribution and supply systems, both topological and operational, are evident and many of the constraints in the electric case can be applied directly to the hydraulic case (continuity of flow, demand profiles, topology,...). Therefore, the success of PLANETS justified to a large extent the application of CLP-modelling techniques to water-distribution control and management. The *relevance of CLP in modelling flow networks* and combining various aspects of network operation in one single problem solver, had been clearly demonstrated. This achievement has become one of the most important lines of pioneer research of the UPC group.

[1] Commercialized by Cosytec, SA, Paris, France.

Domain Variables

In PLANETS, the constraints defining the problem were constraints between three types *of (domain) variables*: *temporal*, *topological* and *electric*. These three problem dimensions have their equivalent in CLOCWiSe.

The *temporal* domain variables represent the time points on which *maintenance tasks* are initiated. This temporal dimension is also present in CLOCWiSe. However, the concept of '*task'* has been generalised to whatever '*operation'* that can be carried out on the water-supply network. Hence operations include maintenance, repair, emergency actions, valve switching and pumping operations.

The *topology* of a power-distribution network was completely defined through a set of binary domain variables, representing the states of all switches at all time points. Again, this concept has been generalised in the water case. Instead of talking about 'topology', for a water-supply network we will utilise de more general concept '*configuration'*. Configuration includes the states of all *valves* at all time points, as well as the 'states' of all *pumping stations* and *boreholes*. Both cases are represented by a set of variables with domains denoting the possible configurations. Evidently, binary domains are not sufficient anymore and for each element multiple configurations are possible.

The third, *electric*, problem dimension in PLANETS was made up of variables representing the flow of current in all branches at all time points. In CLOCWiSe, this dimension corresponds to the *flow* of water through the connections of the network.

Probably the most fundamental difference between the two cases is the appearance of a new dimension in the hydraulic case: the *storage* of water in tanks and water towers. In the electric case no storage was possible in network nodes. As a consequence of this difference, a new set of domain variables had to be introduced to represent the *volume* of water in each one of these storage elements at all time points.

Constraints

We will now describe the different kinds of constraints which exist among these variables. In PLANETS, the following constraints were treated: *isolation* constraints, *task-resource* constraints, *task-precedence* constraints, *consumer-service* constraints, *current-continuity* constraints, *switch-behaviour* constraints, *network-radiality* constraints, *overload* constraints, *energy-demand* constraints and *due-date* constraints. A full description of these constraints as well as of their relations, can be found in [Creemers 95].

In CLOCWiSe, most of these constraints (or their equivalent) can be found back. Additionally, CLOCWiSe treats much more constraints, and, being conceived as an open and modular system, new constraints are added to it regularly. Hence the following bullet list should not be considered a comprehensive list of all constraints that can be treated, but is rather meant to give the reader an idea of the diversity of aspects considered and, in some cases, of the expressiveness of the declarative CLP language. Examples of CHIP code given here are merely intended for illustration purposes, since it would be out of the scope of this paper to elaborate on syntax and semantics of the language primitives.

- **Consumer-demand Constraints**: The water demand at a consumer node corresponds to a predicted profile for the planning horizon. Such profile is a list of values for each consumer node and for every time slot. These values are used directly to constrain the initial domains of the flow variables in the connections leading to the consumer.
- **Flow-continuity Constraints**: On all non-storage nodes, except production points, sources and consumers, the continuity laws must hold at all times. These constraints state that the sum of all incoming flows in a node must equal the sum of all outgoing flows. They are linear expressions between flow domain variables.
- **Flow-continuity Constraints with storage**: These constraints say that the sum of all incoming flows in a storage node must equal the sum of all outgoing flows plus the accumulation of volume in the node. They are linear expressions between flow variables and volume variables.
- **Configuration Constraints**: These constraints relate the configuration variables of a configurable element (valve, borehole, pumping station) to their corresponding flow variable, at a particular time point. In CHIP, they are straightforwardly coded using the built-in `element/3` constraint, e.g.

  ```
  element(Conf, [0,120,330,410],Flow),
  ```

  where `Conf` is a configuration variable with domain {1,2,3,4} and `Flow` is the flow variable. Whenever the domain of any of the two is reduced, the domain of the other one is updated consistently.
- **Maintenance Constraints**: These constraints relate task start-time variables with the water-network variables. They state that on the particular element under maintenance flow equals zero during task execution, and are implemented by means of an extension to the built-in `element/3` constraint, forcing a number of elements in a list of flows (of the maintained pipe) to be zero, starting at a position indicated by the temporal domain variable, i.e.

  ```
  cw_element(Start,Duration,Flows,0),
  ```

  where `Start` is the start-time variable of the task, `Duration` is its duration and `Flows` is the list of domain variables representing flow through the maintained connection at all time points.
- **Resource Constraints**: For any task requiring resources (manpower, equipment, etc.), the available amount of the particular resource must be respected at all times. This constraint has its implications on the possibilities to carry out jobs concurrently. It is enforced by the use of CHIP’s `cumulative/n` built-in, e.g.

  ```
  cumulative(Starts,Durations,Resources,…),
  ```

  where `Starts` and `Durations` correspond to lists of start times variables and durations of the tasks, and `Resources` is the list of requirements of a particular resource.

- **Temporal order Constraints**: Such constraints can be stated to enforce a specific temporal order between tasks. They are straightforward uses of inequality constraints between temporal variables.
- **Capacity Constraints**: The minimal and maximal values for either flows through network connections or volumes in storage nodes, can be specified directly by constraining the initial domains of the corresponding variables, at every time point.
- **Due-date Constraints**: Any job must finish before its due date. This is enforced by constraining the initial domain of start time variables.
- **Number-of-changes Constraint**: This constraint can optionally be stated to restrict e.g. minimal or maximal number of pump on-off switchings or valve-position changes. Use of this constraint will produce more 'practical' schedules where pumps do not have to be switched on and back off every hour. In a traditional approach it is very difficult to enforce this kind of behaviour. In CHIP it comes down to the use of the global constraint **`sequence/5`**, e.g. (only meant for illustration purposes):

```
sequence([Min,Max,2],VarList,C,[[[range,2,#>=,[1]]]],all),
```

- **Flow-rate-change Constraint**: This optional constraint can be used to restrict the rate of change of flow through a network element. From an operational-control point of view this is an important constraint. However, classical approaches experience great difficulty in expressing it. Again, it can be implemented using **`sequence/5`**, e.g.

```
sequence([0,0,Seq],FlowLs,C,[[[range,Seq,#>=,[Range]]]],all),
```

To conclude we merely mention the rest of the constraints treated in CLOCWiSe:

- **Electricity Tariff Constraints**
- **Mixing Constraints**
- **Water Purchase Constraints**
- **Borehole Operational Capacity Constraints**
- **Tank Target Level Constraints**
- **Leakage Repair Constraints**
- **Calendar Constraints**
- **Task pre-condition Constraints**

The interactions between the above types of constraints and the domain variables on which they act are graphically represented in Figure 2.

Constraint Propagation

The figure could be seen as a high-level representation of the system's *constraint network*. This is a network in which variables are nodes and constraints are the links between them. Links not necessarily need to be binary. Indeed, many of CHIP's most important constraints are so-called *global* ones (like **`cumulative/n`** and **`sequence/5`** above) which involve many variables at once. The consistency algorithms behind each one of these links will enforce that the domains of their attached nodes (variables) are kept at all times compatible. In this way, when new information about a variable becomes available (i.e. as a reduction of its domain), this information will propagate through the constraint network and domains of other variables will be reduced. This *constraint propagation* process will continue until all domains are consistent again. This is where the LP part comes in to make some logical choices about further domain reductions so that the propagation can go on, until all variables are instantiated and a solution is reached which satisfies all constraints. This phase is called *variable labelling*.

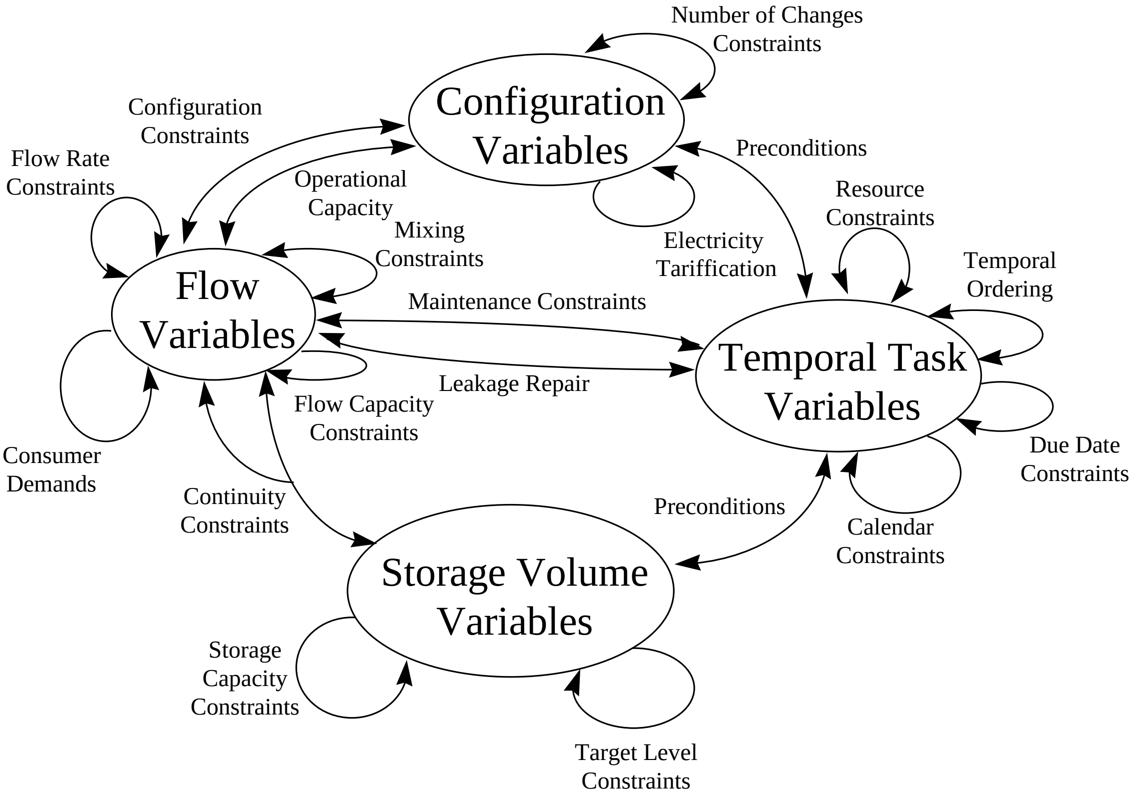

Figure 2. A high-level representation of the constraint network.

Optimising the solution

In most practical situations, we don’t just want a solution satisfying all constraints, but we rather want to find the *optimal solution*, with respect to some *objective function*. In particular, in CLOCWiSe’ scheduler (see paragraph 5.1) the main objective is to *minimise pumping cost*. With this objective in mind, the CLOCWiSe logic program part in charge of variable labelling, implements various optimal strategies to home in on the optimal solution as soon as possible. The classical approach in finite domain CLP programs is to apply a *branch-and-bound* algorithm, iteratively adding additional constraints on the objective variable, so that ever better solutions are obtained in subsequent iterations. The CLOCWiSe scheduling engine combines this approach with, among others, a credit-based *limited discrepancy search* [Harvey 95].

4. The Software

4.1 Architecture

The CLOCWiSe overall software architecture is depicted in Figure 1. The overall system is centered around the *Network Configuration Management and Control* module, which implements the constraint engine described in paragraph 3.3. The engine is fed with a list of maintenance or repair tasks to be carried out on the network and with the demand predictions for the chosen planning horizon. Maintenance is typically scheduled on a weekly basis. Schedules can be sanctioned by authorized users for implementation and dispatching of resources. Such sanctioned schedules convert into additional constraints for any subsequent run of the constraint engine on overlapping planning horizons. In this way, week schedules are made consistent with the higher-precision day schedules. An *Emergency Handling* module facilitates the inclusion of urgent repair tasks, specification of degraded performance of certain network elements, and the re-scheduling of previously sanctioned tasks and operations. Finally, a specialized module provides *estimations* of water quality model parameters and of the actual quality state, based on measurement information and/or a-priori information.

4.2 Graphical User Interface

All modules and functionalities are embedded in a uniform, innovative Graphical User Interface (GUI). The GUI is an essential part of the overall CLOCWiSe philosophy. In the first place, it provides the user with a *generalised network editor*. This editor was designed to transfer the expressiveness and many other advantages of the declarative CLP paradigm used, all the way to the end user of the software. The network editor allows the user to construct not just a model of the physical network, but of a generalised network, including non-physical elements and connections, such as *tasks, resources, assignments, emergencies, calendars, influences, relations*, etc. Moreover, these items known to CLOCWiSe are organised in separate libraries which can be loaded into the editor. The interface provided by such library elements is well-defined and new elements and connections can be written easily, specifying their full behaviour and intrinsic constraints. As a result, the user has maximal flexibility in specifying his/her specific problem, and what he/she is actually doing in the network editor is a graphical form of constraint programming. Figure 3 shows the main screen of the GUI and gives an example of how a maintenance task and its resources are included as nodes and connections of the generalised network. Apart from this main screen the GUI offers a collection of Gantt charts, graph windows and tables.

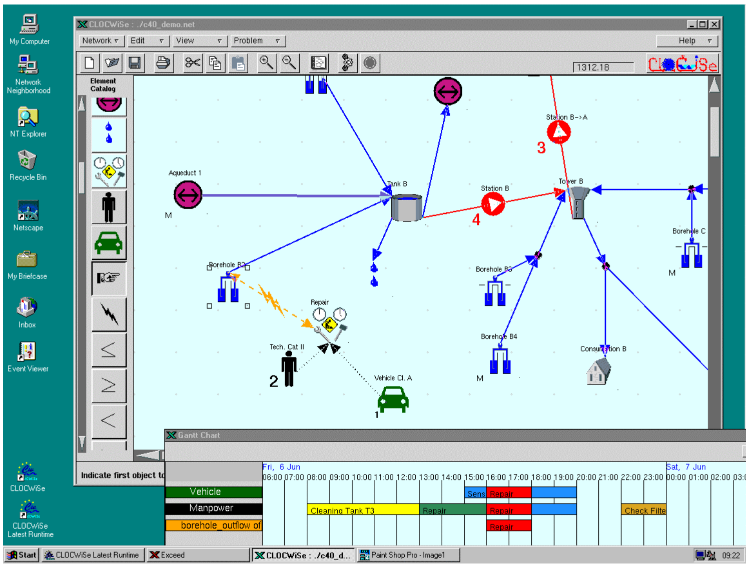

Figure 3. The Graphical User Interface

5. Applications

5.1 Scheduler

The CLOCWiSe scheduler receives as input a particular water supply network, described in the GUI as a set of *physical and operational constraints* together with a list of (maintenance) tasks to be carried out during the specified planning horizon. Tasks are described in terms of their *earliest-starting-time*, *latest-finishing-time*, *priority*, use of *resources* and the *influence* the task has on physical network operation, both during task execution and afterwards. With such complete description of the problem in terms of constraints, the problem solver initiates a search for the schedule satisfying all of them at minimal cost. In the present version of the software, minimal pumping energy cost, taking into account the actual electricity tariff structure, is used as the objective function. Other alternative objectives, such as minimizing the usage of a particular resource or the difference with target levels, are being added. For a network corresponding to a substantial subnetwork of a representative European city, containing around 10 possible extraction points and several storage reservoirs, the scheduler comes up with an optimal maintenance and operational schedule in less than a minute. The solution schedule is presented in the form of separate graphs for each network element and a global gantt chart including tasks and pump operations (see Figure 4 and Figure 5).

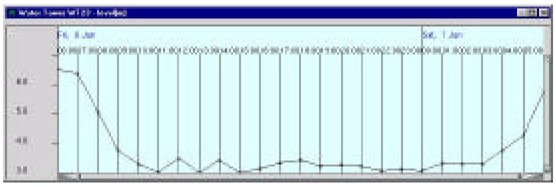

Figure 4. A water tower level graph.

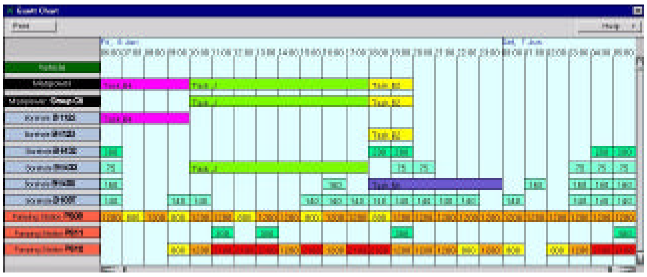

Figure 5. A solution Gantt chart.

5.2 Estimator

A flow-controlled mixing activity is carried out in one of the reservoirs of the above network in order to obtain desired concentration of conductivity in the reservoir. The conductivity was estimated on-line based on the input and output noise corrupted conductivity measurements and noise free flow measurements. The results are illustrated in Figure 6. The envelopes bounding the output conductivity (bold dashed) together with the estimated value (solid) are compared against the direct measurements (dotted) showing a valuable reduction of the measurement error by the estimator. The estimation without the output measurement was also carried out. The estimator operates then as a simulator of the output conductivity and can be viewed as a generic soft sensor. The results obtained demonstrated that in this application the hard sensor would be replaced by the soft one reducing the cost, increasing the instrumentation reliability and more importantly allowing to predict the conductivity.

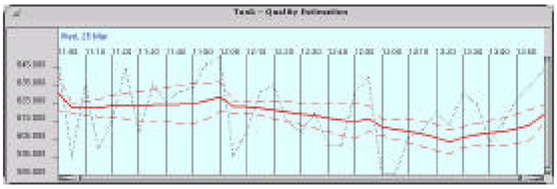

Figure 6. Estimation of conductivity.

6. Conclusions

CLP provides the technology for a new form of decision support for integrated management and operational control that overcomes some of the drawbacks of simulation-based techniques. The CLOCWiSe application proved that this concept can be implemented and works effectively when applied to real-life systems.

A very strong aspect of the technology is that the solving process is based on reduction. This supports an approach of specifying the goals to be achieved, and the possibilities and limitations of the system during the design phase and optimizes the interactions between them, in a very close manner to the human way of problem solving. Moreover, with CLP it is often the case that complexity not necessarily decreases efficiency of the solver, since including constraints from different angles can help reducing the search space. A third benefit is the natural way CLP offers to deal with uncertainties and incomplete information, as evidenced by the estimation module. Of course, a lot depends on the information available. If there is too little information, lack of pruning possibilities may produce long and unfeasible solving times. If sufficiently informative constraints are available, the solution time may become very short. A still weak point of CLP is that few explanation can be given on which constraint is the main responsible for failure in the case of an over-constrained system. This is the topic of intensive research carried out by the CLP community.

One of the main aspects of decision making is strategy development. Defining the way to find a solution for a problem is creating the backbone of the solution process and determines the quality of any solution. It is in this aspect that CLP offers great support not only through its flexibility and suitability for strategy definition, but especially through its potential to build these strategies dynamically during the solving process.

Through the use of these features CLOCWiSe offers unique functionality to the drinking-water world, like flexibility in specification, fundamental integration of system configuration and maintenance task planning, estimation functionality yielding robust bounds for hydrodynamic and quality parameter values and flexibility for schedule refinement for different time-scales. We believe that similar functionality could not have been achieved with traditional approaches and it shows that new forms of DSSs, closer to the human way of problem solving and decision making, can be constructed.

7. Bibliography

[Brdys 94] Brdys, M.A., Ulanicki, B., *Operational Control of Water Systems: Structures, Algorithms and Applications,* Prentice Hall, New York, London, Toronto, Sydney, Tokyo, 1994.

[Brdys 99] Brdys, M.A., Robust Estimation of Variables and Parameters in Dynamic Networks, *Proc. of the 14th IFAC World Congress,* Beijing, July, 1999.

[Brdys 99a] Brdys, M.A., Lisiak, Z., Robust on-line monitoring of mixing quality in water supply networks. *Proc. of the A.S.C.E. 26th Annual Water Resources Planning and Management Conference,* Tempe, Arizona, June 6 - 9, 1999.

[Creemers 95] Creemers, T., Ros, L., Riera, J., Ferrarons, C., Roca, J., Corbella, X., Constraint-based Maintenance Scheduling on an Electric Power-Distribution Network, in *Proceedings of The Third International Conference and Exhibition on Practical Applications of Prolog,* Paris, April 1995.

[Creemers 98] Creemers, T., Ros, L., Riera, J., Roca, J., Ferrarons, C., Smart Schedules Streamline Distribution Maintenance, in *IEEE Computer Applications in Power,* Vol. 11, Nr. 3, pp. 48-53, 1998, IEEE Power Engineering Society.

[Dincbas 88] Dincbas, M., Van Hentenryck, P., Simonis, H., Aggoun, A., Graf, T., Berthier, F., The Constraint Logic Programming Language CHIP. In *Proceedings of the International Conference on 5th Generation Computer Systems*, ICOT, 1988.

[Harvey 95] Harvey, W.D., Ginsberg, M.L., Limited Discrepancy Search, *in Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Canada, August 1995.

[Van Hentenryck 89] Van Hentenryck, P., *Constraint Satisfaction in Logic Programming*, Logic Programming Series, MIT Press, Cambridge, MA, 1989.

[Wallace 96] Wallace, M., Practical Applications of Constraint Programming, in *Constraints, an International Journal*, Vol. 1, pp. 139-168, 1996, Kluwer Academic Publishers, Boston.

Corresponding authors

For further information, please contact:
Mietek A. Brdys, School of Electronic and Electrical Engineering, The University of Birmingham, Birmingham B15 2TT, UK. Phone: + 44 (0)121414 4354. Fax: +44 (0) 121 414 4291. Email: m.brdys@bham.ac uk
or:
Tom Creemers, Institute of Robotics and Industrial Informatics (UPC/CSIC), Av. Gran Capità, 2-4, 08034 Barcelona, Spain. Phone: +34.93.401.57.91. Fax: +34.93.401.57.50. Email: creemers@iri.upc.es
or:
Hans Goossens, WL|delft hydraulics, PO Box 177, 2600 MH Delft, The Netherlands. Phone: +31.15.285.84.77. Fax: +31.15.285.85.82. Email: hans.goossens@wldelft.nl