

# APPLICATION OF ARC IN SYSTEM DESIGN

Hylke van Dijk      Koen Langendoen  
Henk Sips

Faculty of Information Technology and Systems  
Delft University of Technology, The Netherlands  
{hylke,koen,henk}@ubicom.tudelft.nl

## Abstract

Current and future mobile communication systems require the combination of high performance technology from very diverse engineering disciplines. Consequently, these systems are complex by nature, which complicates system-wide Quality of Service optimisation. In this paper we consider system-wide optimisation of resource distribution within the framework of a given (static) architecture but with components that do support multiple modes of operation. We take into account variable type of applications, quality requirements, resource availability, and external context. In this paper we use the concepts of *Adaptive resource contracts* (ARC) to capture and exploit the capabilities of individual components in the system. A relevant case study demonstrates the added value of the ARC concepts for system design. Eventually the method yields a specification of individual system components and an indication of the optimised overall QoS.

## 1 INTRODUCTION

Recently, a number of demanding applications have been proposed in the emerging market for mobile communications. The high performance systems that implement these applications are undeniably complex and, consequently, designing them is a complex task. The functional structure can be borrowed from existing systems and individual components can be upgraded to higher performance, but the mutual tuning of all high performance components is not a trivial task. In the end, all components must cooperate to compose a high performance system. Solving the design problem with brute force is undesirable, since adding high performance components sincerely increases the overall complexity.

In this paper we address the efficient distribution of resource budgets in a system with a given structural description. We assume that each system component has a well-defined functionality, but in order to do optimisation, the system components must expose a certain amount of flexibility: they must be adjustable. Also, close collaboration between components is necessary in order to achieve the common goal. Ideally, the collaboration is handled by interfaces that capture options and expose opportunities in a mutual language. The problem is even more complex because components in a mobile communication system

originate from very diverse engineering disciplines. Still, control interfaces are required to modify the behaviour of components so as to optimise the over-all system performance.

Mobile communication systems operate in multiple settings. Every setting poses its own constraints on the system and thus influences the way a component is applied. A scenario combines a range of these settings. By doing a careful analysis of a scenario a design specification can be derived for every component in the system. For instance the flexibility of a component can be quantified.

In the rest of this paper we present our system design philosophy in the context of mobile communications. The method is applied in a case study from the Ubicom programme. Results are discussed for a set of example scenarios.

## 2 DESIGN PHILOSOPHY

Our system design philosophy developed in the Ubicom [4] research programme for mobile visual augmented reality. Ubicom considers so-called context-aware systems for mobile communications. The challenge of these type of systems is that their functionality changes with environment parameters. For instance, location dependent information is provided to a user (path finding) or the applied radio technique adapts to the user's actions and circumstances (infra red indoor and GSM outdoor). Ubicom is at the high end of the class of context-aware systems. Augmented-reality supports applications that render virtual objects on a see-through display and connect them one-on-one to real-world objects.

Augmented reality pushes the limits of current technology: these systems inherently must be capable of operating in a wide range of settings. A setting specifies more than the context parameters as used by the application. It also includes, e.g., the way a user will use the system. The mobility aspect is another source of context changes. At one time the channel is crystal clear while at other times interference disturbs proper reception. Without high performance components we can never design a system that approaches the high expectations.

A commonly accepted strategy is to design for worst-case behaviour. However, due to the wide range of settings (context specifications) this is not an option. At some point a trade off must be made and therefor components must be flexible. Designing flexible high performance components is a specialists task. System design can exploit the expert knowledge if we capture it by using sensible abstraction of behaviour and options a component can offer. Making appropriate abstractions requires an interactive procedure. This procedure starts with requesting a rough performance indication from a component. The request is in terms of the abstraction. In return the component offers its capabilities in the requested range, also in abstract terms. In the interest of the entire system one can select one of the offered options or request an alternative offer.

It is widely accepted to structure complex systems hierarchically. We recognise components that act as a *client* using lower level components, operate as a *server* supporting components up the hierarchy, or both. The client-side of a component generates a workload and the server-side processes an induced workload. Applied in the hierarchy, the specification of the enforced workload on the server-side of a component and a specification of the retrieved performance at the client-side specifies the *context* in which the component operates. The context yields the necessary information for a component to specify its behaviour. A scenario

combines a range of settings for the entire system and, consequently, combines a set of contexts for individual components.

The context information corresponds to a specification of interface parameters of which all parties involved have mutual understanding. Context information therefor acts as the control interface between components and the interface parameters bridge fields of expertise. As an example, a source coder uses a transceiver to transfer data. Implementation details of the transceiver are not of any concern to the source codec, but its behaviour is. Similarly the transceiver is not concerned with the implementation details of the source coder, details about the expected performance and induces workload are. One may expect to find interface parameters like throughput, bit error rate, latency, and power consumption at this interface. If necessary more parameters can be used to fine tune context information for both parties but usually the behaviour can be captured in couple of parameters

## 2.1 ARC

In [2] we introduced a generic scheme, named Adaptive Resource Contracts (ARC), for doing run-time adaptations in hierarchical systems. The same concepts can be used statically to analyse a system under design and derive design specifications for individual components, given the set of scenarios. Figure 1 outlines the ARC concepts. Components can only optimise their range of operation modes if they can derive their context. The process starts with a client issuing a *request* to a server. A request is a partial specification of the expectations a client has about the performance of the server. The server responds with an *offer* stating possible performance options. The client now, can either select an appropriate option or generate a new request. Although Figure 1 does not show any hierarchy, the process also works in a hierarchical setting. The server may act as a client itself and send a request to lower level service providers.

Offers and requests are specified in interface parameters and corresponds to an actual internal mode of operation of the server. Hence the generated offer is an exposure of the component's *operation space*. The received request and offer form the necessary context information of a component. Given this context, the component can analyse the consequences of matching internal setting with offered options; the component can generate an offer itself and thus expose its operation space. To prevent an explosion of the number of options in the operation space, components filter out unattractive options. Service providers at the lowest level in the hierarchy can generate offers directly, whereas clients at the highest level can immediately select the most appropriate performance option from the received offers. The selected option is referred to as a contract. The contract establishes the internal settings of the server and in turn it defines the contract with lower level service providers. Since we allow requests, offers, and contracts to change with a change of context, we dubbed the concept Adaptive Resource Contracts (ARC).

When a scenario comprises a wide range of settings, run-time adaptation will be necessary for almost all components in the system. The exposure of operation spaces necessary to do system design are equivalent to the ones used in ARC. We can reuse the design time results at run-time. As indicated, the operation space is based on internal parameter settings. The parameters include details of the implementation of a component. The operation space can be an abstraction of a set of real time implementations, or it can be the result of a methodology for design space exploration [3].

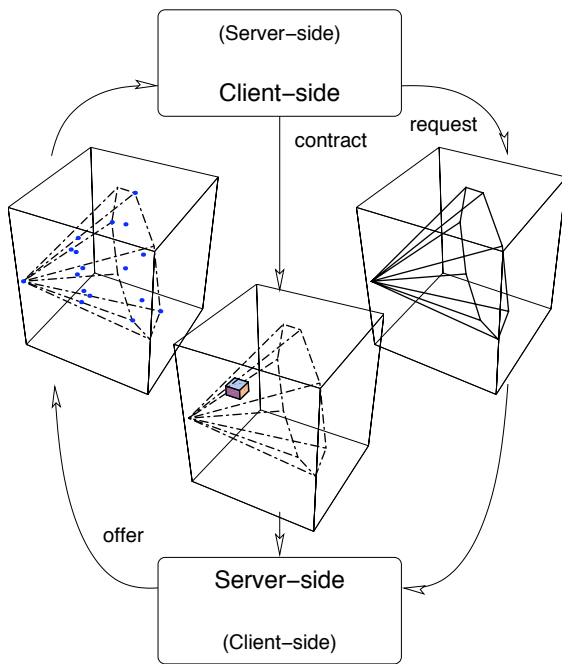


Figure 1: ARC in a hierarchical setting.

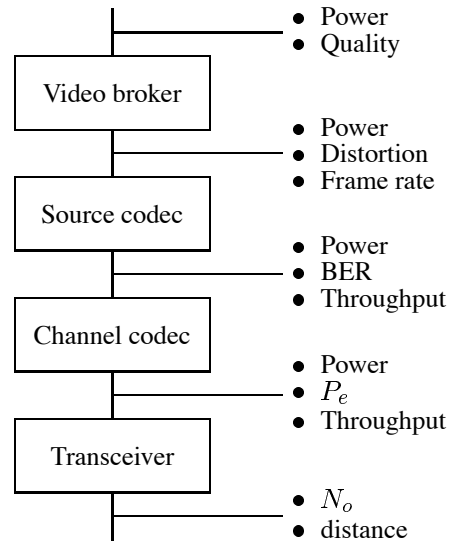


Figure 2: Model component definition and interface parameters.

### 3 MODELLING

To demonstrate our interactive system design philosophy we present a (linear) hierarchical model of a wireless multi media communication system. The model is highly structured. Each component has client-side workload generation functions, server-side operation space exposure functions, and optimisation filters. The model evaluates and combines the exposed operations spaces of individual components. Since the system will be battery powered, controlling the power dissipation is the glue in the design. Utilisation of other resources, like chip area, weight, etc., although important, is not considered in this case study. Components implement a strategy for filtering out irrelevant options based on a locally defined quality over power metric. The application involves two mobile devices that communicate video information. Users of this system get the opportunity to control quality and power dissipation. They can watch high quality video for a short while or lower quality for a longer time. The components in the system collaborate to achieve the overall system goal: bringing the best possible quality at controlled expenses.

#### 3.1 STRUCTURAL MODEL

Figure 2 shows the control structure of the video broker application. We chose a straightforward partitioning in components. The broker is the interface to the user and uses a source codec to transmit video with a controlled quality and resource utilisation. In turn, the source codec uses services from the channel codec to transport the bit-stream. Finally the channel codec issues a protected bit-stream to the transceiver that modulates and transmits the bit-stream over the mobile channel. The parameters at the interface between two components are given in Figure 2. These parameters have well defined meanings for both parties involved, although their respective interpretation (i.e., translation to internal parameters) may differ.

Mapping of functionality is straightforward. We assume a single CPU system. The CPU and the transmitter front-end are the only components in the system that draw a current directly from the battery.

### 3.2 COMPONENT MODELS

The model implements the interactive procedure with requests, offers, and contract as explained in Section 2. Unfortunately here we can not describe each component in full detail. The interested reader is referred to [7]. Here we present an excerpt.

**CPU SYSTEM** After [5] we model the CPU system with full support for frequency scaling and an additional scheme for voltage scaling. The dissipated power is proportional to the requested number of cycles to be processed. The slope of the proportional relation has two modes: a low voltage for a moderate number of cycles and a high voltage for the rest. Figure 3 shows the diagram. A shortcoming of this first-order approximation is that we do not take into account the utilisation of memory by the algorithms. It is a known fact that data transfers to and from memory, for instance, is a power hog.

**TRANSCIVER** The physical channel is part of the context of the transceiver. The background noise  $N_o$  (including interference) and distance between mobile units plays a crucial role in determining the operation space of the transceiver. Both parameters have a negative influence on the perceived error probability ( $P_e$ ) in the transceiver's operation space. Figures have been calibrated using the measurements from [1].

**CHANNEL CODEC** The channel coder adds redundant bits to enhance the error probability  $P_e$  as offered by the transceiver. The resulting bit error rate (BER) is specified in the channel codec's operation space. In our model we use a  $(n, k)$  block code with a performance comparable to the theoretical so-called sphere package bound. To calibrate the induced workload on the CPU system we use a reference implementation of a Reed-Solomon decoder on a Pentium processor.

**SOURCE CODEC** The source codec implements a so-called progressive coding technique. Again, we use a theoretical model for relating the perceived distortion and the number of correctly transmitted bits in a block. In [6] an exponential relation between encoded variance and transmitted bits is derived. The actual value of the exponent depends on the content of the video. We assign a larger exponent to a video sequence with highly moving content which relates to frame rate in our model. The normalised exponential relation is shown in Figure 4 for a range of video categories from still picture to the notorious MTV video clip. The latter one, being very dynamic, has almost a proportional relation between transmitted bits and encoded variance. The codec is assumed robust: transmission errors do not prevent the decoding process, although they reduce the effective encoded variance and, thus, increase the distortion.

**VIDEO BROKER** The video broker defines the quality measure offered to the user based on the distortion and frame rate in the source codec operation space. We assume a desired frame rate of 25 fps. Lower frame rates increase the perceived distortion and consequently decrease the quality metric in the operation space of the broker. The video broker does not impose a workload on the CPU system.

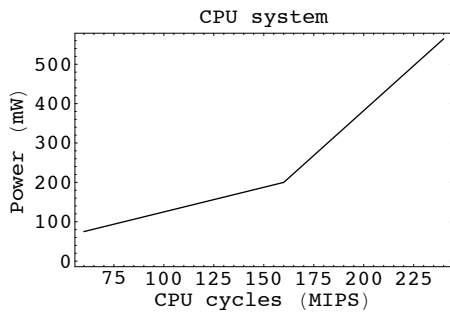


Figure 3: Operation space of the CPU system.

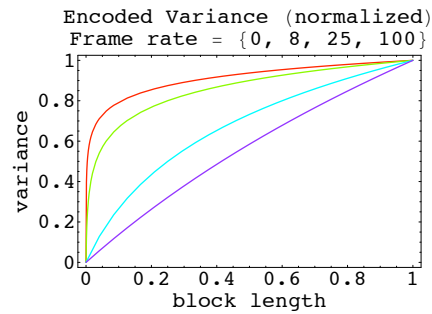


Figure 4: Optimisation space of the source codec, see [6].

## 4 DISCUSSION

We evaluated a range of scenarios using the model from the previous section. All scenarios have a fixed interference noise level set to that of a cellular network with 18 interfering neighbouring cells and a cell radius of 400 meters. The scenarios cover a mobile-to-mobile distance ranging from 1 to 600 meters and the system offers perceived overall video quality in the range of 0 to 0.9. Unfortunately high quality at long distance is prohibitively costly; the system dissipates up to 400 W.

Figure 5 shows the combined operations spaces for different mobile-to-mobile distances. As can intuitively be expected, increasing the perceived image quality will increase the dissipated power by the system and, similarly, the power dissipation rises with the increase of the distance between mobiles. Although Figure 5 shows the capabilities of the system as a whole, it does not reveal design constraints for individual components. The model can provide them, as we will explain by considering some example scenarios.

In the first scenario we require the system to achieve a fixed perceived overall image quality, initially set to 0.6. The diagram in Figure 6 shows the power budget distribution over the system components. Already at a distance of 100 m the transceiver dissipates 12.5 W. Closer examination shows that this is mainly due to extensive base-band processing. With the increase of the distance between mobiles more complex modulation schemes are required, although the raw throughput is fixed at 12 Mbs. Channel coding is only necessary at long distance, at close range it can be switched off. The source coder has, in this scenario, a fixed mode of operation. When we reduce the demands on the quality offered by the system to a level of 0.25, the component specifications obviously change. Still covering a distance from 1 to 600 m, the scenario is now best served with a fixed mode of operation for the source coder, the channel coder, and the base-band processing of the transceiver. The single component that is required to adapt to changes in the distance between mobiles is the transmit power of the transceiver.

In a second scenario we fix the mobile-to-mobile distance at 250 m. Figure 7 shows the number of cycles induced by the respective component on the CPU. The diagram demonstrates the known fact that it is generally not a good idea to implement base-band processing on a general purpose CPU. For long distances the base-band processing requires up to 700 GIPS with which it offers a raw throughput of 28 Mbs. Figure 8 shows the relative throughput per component for the same scenario with a fixed distance of 250 m. The transceiver relates the required throughput to maximum requested throughput, the channel codec

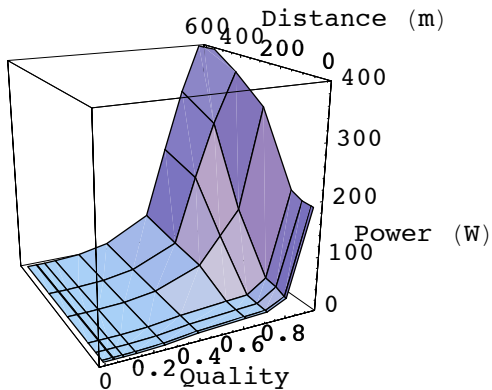


Figure 5: System-wide operation space for various mobile-to-mobile distances.

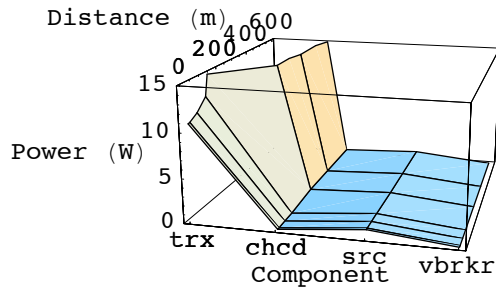


Figure 6: Power budget distribution per component with  $Q = 0.6$ . [trx: transceiver; chcd: channel codec; src: source codec; vbrkr: video broker]

shows the redundant portion of the stream and the source coder the normalised encoded variance. For this scenario all components have to implement their entire range of possible modes of operations. As an example the transceiver's relative throughput runs, with the increase of the overall system quality, from 1.5 to 100 percent. The source coder encodes 10 to 90 percent of the image variance. Serious channel coding becomes necessary with the increase of quality demands.

Reaching the bounds of a posed request, as is the case for the throughput in the second scenario, may indicate a possible bottleneck that can hamper us in finding the system-wide optimum. Bottlenecks are simply removed by relaxing the request. In this case that is not necessary since we are at the extreme end of the system-wide request range for perceived image quality.

## 5 CONCLUSION

The future generation of systems for mobile communications have high performance demands. Consequently, these systems are complex. Even more so, because high performance components from very different engineering disciplines must closely collaborate in a mobile environment with changing conditions. To handle different context settings, system components must support multiple modes of operation. The question remains how many operation modes are required and which mode to select when. To answer these questions we outlined an interactive design philosophy that is based on requests and offers. In this method components receive a request with expectations about its behaviour. In return it replies with an offer that specifies its behaviour based on its current context. It is important that offers contain multiple options to support mutual optimisation between components. With a recursive application of requests and offers, the system eventually selects the most appropriate option while taking into account the system-wide scenario.

The design philosophy has been implemented using the concepts of the adaptive resource contracts (ARC) framework. ARC defines operation spaces that expose performance options and captures expert knowledge of components in an hierarchical system. In a relevant

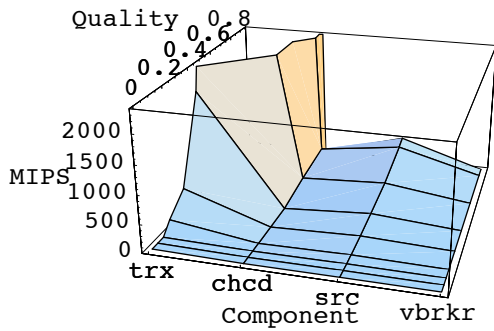


Figure 7: Cycle budget distribution per component at 250 m.

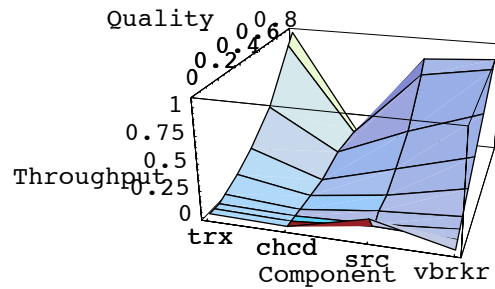


Figure 8: Normalised throughput per component at 250 m.

case study we demonstrate that for various scenarios, components are applied differently. Each setting in a scenario selects for the respective components the most appropriate mode of operation. Through the analysis of a scenario a design specification component is derived for each component. The graphical representation of operation spaces, and derivatives thereof, is certainly an asset.

As an example, the model indicates that implementing the base-band processing of a transceiver on a general purpose CPU is not a good idea. Also we found that for a certain scenario all but one component have a fixed mode of operation. In the most demanding scenario all components require run-time adaptation. Since ARC has been designed for dynamic systems, this framework is a good candidate for implementing the required run-time adaptations.

## ACKNOWLEDGEMENT

This work was conducted within the Ubicom program ([www.ubicom.tudelft.nl](http://www.ubicom.tudelft.nl)) and funded by the TU Delft, DIOC research program. We thank Chris van den Bos, Adrian Bohdanowicz, Maarten Ditzel, and Arjen van der Schaaf for providing valuable input for the case study.

## REFERENCES

- [1] Adrian Bohdanowicz, Gerard J.M. Janssen, and Slawomir Pietrzyk. Wideband indoor and outdoor multipath channel measurements at 17 GHz. In *Proceedings VTC'99*, Sep. 1999.
- [2] H. van Dijk, K. Langendoen, and H. Sips. ARC: a bottom-up approach to negotiated QoS. In *3rd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2000)*, Dec. 2000.
- [3] A.C.J. Kienhuis. *Design Space Exploration of Stream-based Dataflow Architecture Methods and Tools*. PhD thesis, Delft University of Technology, Jan. 1999. ISBN 90-5326-029-3.
- [4] R.L. Lagendijk. The TU-Delft Research Program Ubiquitous Communications. In *Proceedings of the 21st Symposium on Information Theory in the Benelux*, May 2000.
- [5] J. Pouwelse, K. Langendoen, and H. Sips. Voltage scaling on a low-power microprocessor. In *2nd Int. Symposium on Mobile Multimedia Systems & Applications (MMSA'2000)*, Nov. 2000.
- [6] A. van der Schaaf and R.L. Lagendijk. Independence of source and channel coding for progressive image and video data in mobile communications. *Visual Communications and Image Processing (VCIP2000)*, Jun. 2000.
- [7] Hylke W. van Dijk. A case study using Adaptive Resource Contracts (ARC). Technical Report 3, Delft University of Technology, <http://www.ubicom.tudelft.nl>, Nov. 2000.