A Formal Specification of
an Information Processing System in Duration Calculus

Sheila A. Karipel and Dang Van Hung
United Nations University
International Institute for Software Technology
UNU/IIST, P.O.Box 3058, Macau

ABSTRACT
In this paper, we give a formal specification of an Information Processing System (IPS) in Duration Calculus. The IPS we are dealing with is a hard real-time system consisting of two independent subsystems that communicate through several I/O channels with the external environment. It is stated formally that the requirement for the system is met if the earliest deadline first (EDF) scheduler is used and certain conditions on the parameters are satisfied.

Keywords: Formal Specification, Duration Calculus, Earliest Deadline Driven Schedule, Information Systems.

1. INTRODUCTION
Over the years, the rapidly developing technology has made it possible for information processing systems to handle and process substantially more and more information. Though the high speed technology reduced the computation time considerably, a more precise information extraction required the application of highly advanced information processing techniques. These sophisticated and computationally intensive techniques are extremely time-bounded within the systems. This called for an extensive analysis of the design of the systems.

As one of the authors has participated in the earlier development of an information processing system (IPS) which is a hard real-time system, it provided the authors an opportunity to take it up as a case study for demonstrating the powerful benefits of using Duration Calculus (DC) as a formalism for specifying and verifying a real complex system.

Hard real-time systems have been defined as those containing processes that have deadlines which cannot be missed [1]. Such deadlines have been termed hard: they have to be met under all circumstances.

Meeting hard deadlines imposes constraints on the allocation of physical and logical resources of the system at runtime. Typically, the resources are allocated by a scheduling algorithm whose purpose is to interleave the executions of processes in the system to achieve a particular goal which in the case of hard real-time systems is that no deadline is missed.

In this paper, we attempt to specify an appropriate scheduler that controls the execution of the independent tasks and resources within a subsystem. With a formal specification of IPS and the schedulers it used, we are able to verify that the system task set will meet the required deadlines.

The IPS
The IPS as shown in figure 1 is an integration of 2 subsystems viz. P1 and P2 with identical hardware. Each subsystem has a single processing unit and up to 5 I/O channels (shown as bold directed lines). The data is exchanged in the form of “packets” between the environment and the processors at regular intervals of time. The data rates are different for each channel.

The channel C is a bi-directional communication link between the subsystems P1 and P2. However, for convenience, this bi-directional channel has been logically split into two unidirectional communication links, C1 and C2. P1 transmits through C1 and receives through C2 while P2 receives through C1 and transmits through C2.

The explanations that follow are valid only after the processors P1 and P2 have been initialised. It is assumed that the buffers mentioned below can hold only one packet at a time.

Channels and Processors
The processor P1 uses all the 5 I/O channels viz. A1, A2, A3, A4, and C (C1/C2), out of which 2 channels (A3 and A4) are used as dedicated input channels. A1, A2, A3 and A4 are communication links between P1 and the external environment while C (C1/C2) is the common link between the processors P1 and P2. The processor P2 uses only 4 I/O channels (B1, B2, B3, and C) out of which 1 channel (B3) is used as a dedicated input channel. B1, B2, and B3 are communication links between P2 and the external environment.

The channels A1 and A2: The data packets arriving on these channels contain queries regarding the health status of the processor P1.

Once P1 is initialised, it receives a data packet on A1 from the
The task associated with this channel $Ci$ is to extract the information contained in the data packet received, and store the results of the computation in the buffer $Buf_{outi}$. This is done in $\tau_{C1}$ seconds such that $\delta + \tau_{C1} < T_P$, where $T_P = T_{C1} = T_{C2}$ is the periodicity of reception of data by Pi on $Ci$.

The channels $B1$ and $B2$: The data packets arriving on these channels contain queries regarding the health status of the processor P2. Once P2 is initialised, it receives a data packet on $Bi$ ($i = 1, 2$) from the external environment. The interrupt handler for $Bi$ transfers the data to buffer $Buf_{Bi}$. The time taken for handling the interrupt is at most $\delta$ seconds which is the delay incurred due to the processor handling other interrupt(s).

P2 extracts the information from the data packet received and sends out an acknowledgement. All these are done in $\tau_{Bi}$ seconds such that $\delta + \tau_{Bi} < T_P$, where $2 \ast T_{Bi}$ is the periodicity of arrival of data on $Bi$.

The subsystem P2 alternately receives from and transmits to the external environment through $Bi$ in the manner described above.

The channel $B3$: The data packets received on this dedicated input channel of processor P2 contain information, $I_3$. Once P2 is initialised, it receives a data packet on $B3$ from the external environment. The interrupt handler for B3 transfers the data to buffer $Buf_{B3}$. The time taken for handling the interrupt is at most $\delta$ seconds which is the delay incurred due to the processor handling other interrupt(s).

P2 extracts the information, $I_3$ from the contents of buffer $Buf_{B3}$, processes it along with the contents of $Buf_{out2}$ (which contains the information extracted by P2 from the data arriving on channel C2), and puts the result, $I_{out}$, in the buffer $Buf_{P2}$. All these are done in $\tau_{B3}$ seconds such that $\delta + \tau_{B3} < T_{B3}$ where $T_{B3}$ is the periodicity of arrival of data on $B3$.

2. DURATION CALCULUS: A BRIEF SUMMARY

In this section, we give a brief summary of Duration Calculus which will be used as the formalism to specify the IPS in this paper. For more details, readers are referred to [4].

Time in DC is the set $R^+$ of non-negative real numbers. For $t, t' \in R^+$, $t \leq t'$, $[t, t']$ denotes the time interval from $t$ to $t'$.

We assume a finite set $E$ of Boolean variables called primitive states. $E$ includes the Boolean constants 0 and 1 denoting false and true respectively. States, denoted by $P, Q, P_i, Q_i$, etc., consist of Boolean expressions over $E$. A primitive state $P$ is interpreted as a function $I(P) : R^+ \rightarrow \{0, 1\}$. $I(P)(t) = 1$ means that state $P$ is present at time instant $t$, and $I(P)(t) = 0$ means that state $P$ is not present at time instant $t$. We assume that a state has finite variability in a finite time interval. A composite state is interpreted as a function which is defined by the interpretations for the primitive states and Boolean operators.

For an arbitrary state $P$, its duration is denoted by $\int P$. Given an interpretation $I$ of states and an interval $[t, t']$, $\int P$ is interpreted as the accumulated length of time within the interval at which $P$ is present. So for an arbitrary interval $[t, t']$, the interpretation $I(\int P)([t, t'])$ is defined as $\int_{t}^{t'} I(P)(t)dt$. Therefore, $\int 1$ always gives the length of the intervals and is denoted by $\ell$.

The set of primitive duration terms consists of variables over the set $R^+$ of non-negative real numbers and durations of states. In

![Figure 2. The Channel $A_1$ of subsystem P1](image)
A duration formula is an expression formed from terms by using the usual relational operations on the reals, such as equality = and inequality <. A duration formula is either a primitive formula or an expression formed from formulas by using the logical operators ¬, ∧, ∨, →, ⇔, and the chop (see below) and quantifiers ∀, ∃ applied to variables ranging over R+.

A duration formula D is satisfied by an interpretation I in an interval [t', t''] just when it evaluates to true for that interpretation over that time interval. This is written as

\[ I, [t', t''] \models D \]

where I assigns every primitive state a finitely variable function from R+ to \{0, 1\}, and \[ t', t'' \] decides the observation window. So the satisfaction relation has nothing to do with the values of the primitive state assigned by I outside the observation window \[ [t', t''] \]. That is, for interpretations I and I', if

\[ I(P)(t) = I'(P)(t), t' \leq t \leq t'' \]

holds for all primitive states in D, then we can prove

\[ I, [t', t''] \models D \iff I', [t', t''] \models D. \]

Given an interpretation I, the chop-formula D1; D2 is true for \[ [t', t''] \] if there exists a t such that \[ t' \leq t \leq t'' \] and D1 and D2 are true for \[ [t', t] \] and \[ [t, t''] \] respectively.

We give now shorthands for some duration formulas which are often used. For an arbitrary state P, \[ \{ P \} \] stands for \( \bigcup \{ P = \ell \} \) and \( \ell > 0 \). This means that \( P \) holds everywhere in a non-point interval. We use \[ \llbracket \cdot \rrbracket \) to denote the predicate which is true only for point intervals. Modalities ◯, □ are defined as: \( ◯D = \text{true}; D; \text{true}, □D = \neg \neg \neg D. \) This means that \( ◯D \) is true for an interval iff \( D \) holds for some subinterval of it, and \( □D \) is true for an interval iff \( D \) holds for all subintervals of it.

In this paper, we will use the following abbreviations as well.

\[ ◯D \equiv (D; \text{true}) \quad □D \equiv \neg \neg \neg D \]

\( ◯D \) holds for an interval \([a, b]\) if and only if \( D \) holds for some prefix \([a, c]\) of the interval, and \( □D \) holds for an interval if and only if \( D \) has for any prefix of the interval.

DC has a set of axioms about states and rules which is sound and (relatively) complete. The rules are referred to [4] for the proof system of DC.

3. FORMAL SPECIFICATION

In order to write down a formal specification of the IPS, we segregate the fixed properties and the tunable properties of the IPS. The fixed properties refer to the inherent behaviour of the system viz. the effects of the hardware components used. We refer to these hard features of the system as the behavior of the IPS. The tunable properties, as the name implies, refer to the behaviour of the software which can be tuned to meet the requirements most effectively. Hence, the reference to the characteristics of the software processes as the scheduling algorithm. Therefore, it is expected that the behaviour of IPS that controlled by the scheduler will meet the requirements of the IPS.

Our purpose is to write the DC formulas

1. \( \exists_{P} \), to capture the behaviour of the IPS,
2. \( \exists_{R} \), to specify the requirements, and
3. \( \exists_{S} \), to formalise the working of the scheduling algorithm that are valid for any time interval of the form \([0, t]\).

### Specification of the Behaviour of IPS

The behaviour of the IPS is modelled by defining the states of its subsystems, P1 and P2.

Definition of States: Let us define the sets \( Ch_{P1} = \{ A1, A2, A3, A4, C1 \} \) and \( Ch_{P2} = \{ B1, B2, B3, C2 \} \). For each channel \( i \in Ch_{m} \) and \( m \in \{ P1, P2 \} \), we introduce two state variables viz.

1. \( intr_{m}^{i} \in Time \rightarrow \{ 0, 1 \} \) to model the interrupt caused by the arrival of new data on channel \( i \) in subsystem \( m \).
2. \( proc_{m}^{i} \in Time \rightarrow \{ 0, 1 \} \) to model the processing of the contents of buffer \( Buf_{m}^{i} \) by the processor \( m \).

Definition of the Periodic Intervals: For any channel \( i \) of subsystem \( m \), the periodic interval is the time interval starting at \( \alpha_{i}^{m} + k \cdot T_{i}^{m} \) and ending at \( \alpha_{i}^{m} + (k + 1) \cdot T_{i}^{m} \), where \( \alpha_{i}^{m} \) is the arrival time of the first data packet, \( k = 0, 1, 2, \ldots, \) and \( T_{i}^{m} \) is the periodicity of arrival of data. Thus, a periodic interval is a time interval between two consecutive arrivals of data on a channel.

For any channel \( i \) of subsystem \( m \), the interval \([0, t]\) is always expressed by

\[
(\ell \leq \alpha_{i}^{m}) \lor (\ell = \alpha_{i}^{m} \mod T_{i}^{m} ; \ell = T_{i}^{m}) \\
\lor (\ell = \alpha_{i}^{m} \mod T_{i}^{m} ; 0 < \ell < T_{i}^{m})
\]

(1)

That is, for any \( t \) either there is no arrival of data before \( t \), or else, the suffix of the interval \([0, t]\) from the last data arrival before \( t \) is a prefix of a periodic interval.

**Properties of the IPS:** A data packet on channel \( i \in Ch_{m} \) globally \( m \in \{ P1, P2 \} \) raises an interrupt on processor \( m \) at the beginning of the periodic interval \( T_{i}^{m} \). After \( \delta \) (where \( 0 \leq \delta < T_{i}^{m} \)) time units, the processor completes the handling of the interrupt by flushing the data packet into the buffer \( Buf_{m}^{i} \).

Therefore, every \( k^{th} \) periodic interval satisfies the constraint \(^1\) :

\[
[ intr_{m}^{i} ] \land \neg [ intr_{m}^{i} ]
\]

(2)

\(^1\)Note that when \( \neg, \odot \), and \( \odot \) occur in formulas, they have higher precedence than the binary connectives and the modality .
and since it takes at most $\delta$ time units to handle an interrupt, every time interval satisfies the condition:

$$\Box [\text{intr}_m^i] \Rightarrow t \leq \delta \quad (3)$$

When a data packet arrives on channel $i \in Ch_m$ of processor $m \in \{P1, P2\}$, it can be processed only if it has been flushed into the buffer $Buf^m_i$. So the processing of $Buf^m_i$ can take place only after the interrupt caused by the data packet has been handled by processor $m$. Therefore, for any time interval

$$\Box [\text{intr}_m^i] \Rightarrow \neg \text{proc}_m^i \quad (4)$$

The processor $m \in \{P1, P2\}$ handles the processing of the data in the buffers $Buf^m_i$, $i \in Ch_m$, with the help of the scheduler implemented in the system. However, at a time, a processor can process the data of at most one buffer. So, for $j \in Ch_m \land j \neq i$,

$$\Box [\text{proc}_m^j] \Rightarrow \neg \text{proc}_m^i \quad (5)$$

It was mentioned earlier that $P1$ and $P2$ are identical systems, and that channel $C$ (referred to as $C1$ and $C2$ in the context) is the common link between the two systems. It can be seen that $T_{C1}$ and $T_{C2}$ are each equal to $T_{P1} + T_{P2}$. Therefore, $T_{C1} = T_{C2}$. From the description, it is evident that $\alpha_{C2} = \alpha_{C1} + T_{P1}$. This being a property of the IPS, we have

$$(T_{C1} = T_{C2}) \land (\alpha_{C2} = \alpha_{C1} + T_{P1}) \quad (6)$$

Combining the properties of the IPS, we get the specification of the behaviour of IPS during the time interval $[0,t]$ as in Fig. 3.

**Specification of the Requirements of IPS**

Let

$$Bi_{i,Ch_{P1}} = \{A1, A2\}, \quad Bi_{i,Ch_{P2}} = \{B1, B2\}$$

$$Uni_{i,Ch_{P1}} = \{A3, A4\}, \quad Uni_{i,Ch_{P2}} = \{B3\}$$

$$Int_{i,Ch_{P1}} = \{C1\}, \quad Int_{i,Ch_{P2}} = \{C2\}$$

denote the bi-directional, uni-directional, and inter-link channels of the subsystems $P1$ and $P2$.

As described earlier, there is a task associated with each channel in a subsystem. The execution of the task should be completed before the next data packet arrives on the associated channel. In other words, the execution of the task should be completed within the periodic interval associated with the channel. The periodic interval has already been defined in Section 1.

For the bi-directional channels, the processing of data should be completed before half the duration of the periodic interval $T_m^i$ is over. This means that every periodic interval of channel $i \in Bi_{i,Ch_m} \land m \in \{P1, P2\}$ should satisfy

$$\int \text{proc}_i^m = \tau_i^m \quad \ell = 0.5 \times T_m^i \quad (8)$$

For the uni-directional channel $i \in Uni_{i,Ch_m} \land m \in \{P1, P2\}$, every periodic interval should satisfy

$$\int \text{proc}_i^m = \tau_i^m \quad (9)$$

For the inter-link channel $i \in Int_{i,Ch_m} \land m \in \{P1, P2\}$, every periodic interval should satisfy

$$\int \text{proc}_i^m = \tau_i^m \quad \ell = T_m^i - T_m \quad (10)$$

By putting together the requirement for the periodic intervals for different channels $(8),(9),(10)$ and taking into account the representation of the intervals of the form $[0, t]$ via periodic intervals $(1)$, we get the specifications of the requirements to be met by the IPS for the time interval $[0, t]$ as in Figure 4.

**Specification of the Scheduler in the IPS**

The IPS is a hard real-time system integrating two independent subsystems whose behaviour and requirements have been specified in the earlier sections. The existence of an EDF scheduler as an integral part of the operating system within each subsystem is also assumed. Each subsystem has a single processing unit for carrying out the task of processing the data arriving on its multiple I/O channels. The periodicity of arrival of data packets, the computation time required for processing the data in the buffer, and the deadline for completion of the computation task vary with each channel within a subsystem.

Consider channel $i$ of processor $m$. A data packet arrives at time $\alpha_i^m + k \times T_m^i$ time units where $k = 0, 1, 2, \ldots$. It takes at most $\delta$ time units for the newly arrived data to be flushed into buffer $Buf^m_i$. So, in the worst case, the task becomes ready only after $\alpha_i^m + k \times T_m^i + \delta$ time units. Therefore, in the worst case, the periodic interval for the task starts from $\alpha_i^m + k \times T_m^i + \delta$ and ends at $\alpha_i^m + (k + 1) \times T_m^i + \delta$ time units.

After a task becomes ready, it requires a computation time of $\tau_i^m$ time units to process the data in $Buf^m_i$. The accumulated run time of the task should not exceed its deadline $D_m$ which is less than its period $T_m$. The deadline for a task varies with the channel associated with the task as:

$$\begin{align*}
\text{for } i \in Bi_{i,Ch_m}, m \in \{P1, P2\} & \quad D_m^i = 0.5 \times T_m^i \quad (12) \\
\text{for } i \in Uni_{i,Ch_m}, m \in \{P1, P2\} & \quad D_m^i = T_m^i \quad (13) \\
\text{for } i \in Int_{i,Ch_m}, m \in \{P1, P2\} & \quad D_m^i = T_m \\
\end{align*}$$

**The Scheduling Policy**: The tasks to be scheduled of course should satisfy

$$\tau_i^m \leq D_m^i \leq T_m^i - \delta \quad (15)$$

and also by an offset of $\alpha_i^m + \delta$. Leung and Whitehead [5] have defined a deadline monotonic priority assignment that caters for tasks with the time constraint $(15)$. Using the results of Leung and Whitehead, Audsley [2] established schedulability tests for periodic tasks with multiple release times.

The tasks are executed in a preemptive manner: at any instant, the task which has the nearest deadline for completion of its computation is allocated processor time. In literature, this scheduling mechanism is referred to as the earliest deadline first (EDF) scheduler.

Audsley’s tests guarantee the deadlines of periodic tasks which satisfy the time constraint $(15)$. For a given set of tasks, $(i \in Ch_m)$ in a processing unit $m$, the deadline monotonic scheduling is feasible if and only if

$$\sum_i \frac{\tau_i^m + \sum_{j \neq i, D_m^j \leq D_m^i \land \alpha_i^m \leq \frac{D_m^j}{T_m^j} \times \tau_j^m}}{D_m^i} \leq 1 \quad (16)$$
\[ \exists_{\text{env}} = \bigwedge_{i \in Ch_m, \ m \in \{P1, P2\}} \left( (\Box [\text{intr}_m] \Rightarrow \ell \leq \delta) \land \\
(\Box [\text{intr}_m] \Rightarrow [\neg \text{proc}_i]) \land \\
((j \in Ch_m \land j \neq i) \Rightarrow (\Box [\text{proc}_i] \Rightarrow [\neg \text{proc}_j])) \land \\
(\ell \leq \alpha^m) \lor \\
(\ell = \alpha^m \mod T_i^m ; \\
(\land [\text{intr}_i] ; [\neg \text{intr}_i]) \land \\
0 < \ell < T_i^m) \lor \\
(\ell = \alpha^m \mod T_i^m ; \\
(\land [\text{intr}_i] ; [\neg \text{intr}_i] \lor [\neg \text{intr}_i]) \land \\
T_{C1} = T_{C2}^P) \land (\alpha_{C2} = \alpha_{C1} + T_{P1}) \right) \tag{7} \]

Figure 3. Specification of the properties of the IPS

\[ \Lambda_{i \in Bi \land m \in \{P1, P2\}} \left( (\ell = \alpha^m \mod T_i^m ; \\
(\ell \leq \alpha^m) \lor \\
(\ell = T_i^m) \land \\
(\land [\text{proc}_i] = T_i^m ; \ell = 0.5 \ast T_i^m) \lor \\
(0.5 \ast T_i^m \leq \ell < T_i^m) \land \\
(\land [\text{proc}_i] = T_i^m ; \ell \geq 0) \lor \\
(0 < \ell < 0.5 \ast T_i^m) \land \\
(\land [\text{proc}_i] \leq T_i^m) \right) \right) \]

\[ \forall_{i \in Bi \land m \in \{P1, P2\}} \left( (\ell = \alpha^m \mod T_i^m ; \\
(\ell \leq \alpha^m) \lor \\
(\ell = T_i^m) \land \\
(\land [\text{proc}_i] = T_i^m ; \ell = 0.5 \ast T_i^m) \lor \\
(0.5 \ast T_i^m \leq \ell < T_i^m) \land \\
(\land [\text{proc}_i] = T_i^m ; \ell \geq 0) \lor \\
(0 < \ell < 0.5 \ast T_i^m) \land \\
(\land [\text{proc}_i] \leq T_i^m) \right) \right) \tag{11} \]

Figure 4. Specification of the requirements of IPS
Theorem 1 It is proved that
\[\sum_{i=1}^{n} \frac{\tau_i^m}{D_i^m} \leq 1 \implies \exists^{eq}\]

The formal proof of the theorem using the proof system of Duration Calculus is similar to that of Liu Layland’s theorem in [7], and is not presented here.

4. CONCLUSION

This paper sets out the formal specifications of the inherent behaviour of the IPS, the properties of the EDF scheduler, and the requirements of the implementation of the IPS that was taken up for as a case study in DC. By formalising the system that the first author was involved in the design, we have a clear understanding of its implementation. Besides, we are able to verify the system formally. The next step of our work will be the formal proof of the correctness of the system using DC proof assistant in PVS.

References