CanKingdom and Dependable CAN systems

by
Lars-Berno Fredriksson
CanKingdom International Inc.
010110

Introduction

The ISO 11898 protocol Controller Area Network (CAN) is a low level protocol. Any system requires a Higher Layer Protocol (HLP) on top of CAN. It is the HLP that makes some CAN systems highly dependable and others not. The requirements of a dependable system are highly depending on the system itself and on the actual situation. An example is a forest harvester getting its hydraulic oil overheated. If it is running on solid soil, most probably the right thing to do is just to stop. But if it is running on a bog, then it should continue running until it is on firm ground as otherwise it will slowly sink. Another example is an airplane: On the ground it might be safe to shut down the engine but not in the air. Surely you will find a lot of other similar examples for different control systems for vehicles, processes, factory automation, etc., all of them suitable for using the CAN protocol but each having very specific requirements for the HLP. CanKingdom is designed to solve the problem by giving the system designer the possibility to optimize the HLP for his system, still using standard products.

CanKingdom is based on four cornerstones:

1. Clearly separating between module and system level
2. Any communication within a system is predefined
3. Providing a set of building blocks for customized HLPs
4. Postpone decisions as much as possible

The separation between module and system level is the key to dependability. The separation can be made due to the fact that any communication within a control system is predefined. The definition of the communication is the heart of the system. Any module will communicate with other modules according to a well defined set of rules. It is the task of the system designer to set these rules. In a CanKingdom system he can build up these rules by building blocks residing in each module. With such an architecture, it is possible to change the rules at any time, even during runtime. It is like an old kingdom: The King makes the laws for his kingdom and the Mayors in each city sees to it that they are obeyed locally. The mayors are responsible solely to the king and he has got the means to supervise their loyalty. The king may change the laws from time to time to keep his kingdom stable and efficient during times of crises. That is CanKingdom, a powerful tool for the system designer.

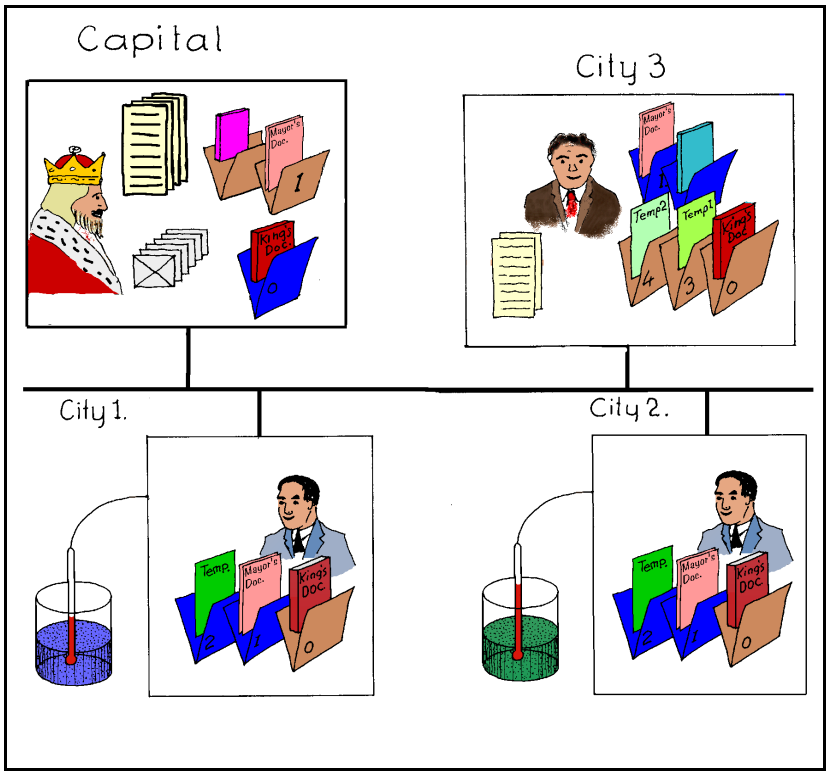
The CAN protocol is used as the low level part of CanKingdom. Some weakness of CAN as pure communication protocol has been pointed in other parts of the Pålbus report as event triggered, non deterministic and not having a stop mechanism or a membership agreement, suffering the babbling idiot problem, etc. By adding control on the system level, these weaknesses can be fully cured and CanKingdom has the building blocks for the remedy. To ensure unambiguity, specific semantics is used in the CanKingdom community. This semantic will gradually be introduced in this paper.

System vs. Modules

The kernel of systems design is organization and management. How to organize and manage a dependable system is highly depending on the system itself and its state. The state of the

system in turn is depending on the states of the different modules in a specific instance of time. A dependable system should always be safe, but in most cases, it is impossible to have a completely safe system during runtime. An aircraft might be considered safe when standing on the airfield with no crew, all systems shut down. But any step taken to finally getting it airborne increases risk. Safety and availability are contradictory requirements and a final system design has to be a compromise between these two. The only one who can make the compromise is the system designer. This can be illustrated by an airplane as a system and an engine as a module. If the engine in a single engine jet fighter faces a serious problem during takeoff, probably the safest thing to do is to eject the pilot and shut off the engine. If it is a civilian plane, the best thing to do is to keep it running and hope for the best, as the alternative to shut the engine will probably kill the pilot anyhow. In a multi engine plane, the best alternative depends on how many of the other engines are running with full power. This shows the importance to distinguish between system responsibility and module responsibility. In CanKingdom, the module designer concentrates on solving local problems and to give selectable alternatives when it can be questioned what is best to do. The system designer organizes the system and makes the selections. In simple systems, the selections might be done once and for all, but usually they have to be done at least at a startup sequence but often also during runtime.

The basic architecture of a CanKingdom system is one supervising module, the Capital, with the supervising software, the King. This is the property of the system designer, the Kingdom Founder. He is totally responsible for the system behavior. The module designer, City Founder, is responsible solely for his module, City, and in this the software, Mayor. The King and the Mayors have each a library of common routines, the King's Document by which the King can distribute his control messages, and the Mayor's Document by which the Mayor can respond. A simple system is shown in fig. 1.



It consists of four nodes, the Capital with the King, two sensor nodes, City 1 and City 2, and one node, City 3, making use of the sensor information. The software is organized in parts. All of the nodes have at least three parts, the King's Document and the Mayor's Document for the system management and a third part for the main application. Entries by CAN messages to

these parts are place in separate Folders, providing accessability and control of the data flow at system level.

Control messages

The King's Document is a set of control messages by which each module is configured to work within the system. They are all carried by one CAN Id, the default being 00_h. The first data byte in the message is the address of one node or a group of nodes. The address 00_h is a broadcast to all nodes. Next data byte is a page number to identify different orders.

Form for the King's Page x:

Document name:	King's Document
Document List:	0.1 Capital / x.0 City
Document Number:	0.1 Capital / x.0 City
Document type:	Transmit (Capital) Receive (City)

Page description.

Page number:

Number of Lines: 8

Data description:

Line description.

The data type of all lines is **Binary.**

Line 0:	City or Group address	
Line 1:	xxxxxxxx	(Page x)
Line 2:	rrrrrrrr	Commands
Line 3:	rrrrrrrr	
Line 4:	rrrrrrrr	
Line 5:	rrrrrrrr	
Line 6:	rrrrrrrr	
Line 7:	rrrrrrrr	

The King's Document is set up as a transmit message for the King and as a receive message for the Mayors. Each Mayor has one transmit message for replies. The CAN Id for this message is the node number plus a "base number" that is distributed by the King during the startup sequence.

Form for the Mayor's Page 0:

Document name: Mayor's Document. Document List: 0.1 Document Number: 1.1 Document type: Transmit		
Page description. Page number: 0 Number of Lines: 8 Data description: City Identification, EAN-13 Code.		
Line description.		
Line 0:	RRRRRRRR	Reserved R = 0
Line 1:	xxxxxxxx	Page x
Line 2:	rrrrrrrr	Response
Line 3:	rrrrrrrr	
Line 4:	rrrrrrrr	
Line 5:	rrrrrrrr .	
Line 6:	rrrrrrrr	
Line 7:	rrrrrrrr	

Startup sequence

For a safe function of the Postal System (communication system) in a Kingdom it is essential that any City acts in a sound and safe way when connected to the Postal System. A City can destroy the communication in two major ways:

1. By using another baud rate than the Postal System.

2. By transmitting a Letter (CAN message) in an Envelope (CAN Id) assigned for transmission of another City.

To avoid destroying the communication by accident, a City will keep silent until its Postmaster (CAN Controller) has received a correct Letter. If the Mayor knows the Base Number of the Kingdom, he announces his presence, otherwise he will wait to be polled by the King.

As the bit-timing register settings can be downloaded by the King, the register settings might be accidentally corrupted during the download procedure. If this happens, it can be impossible to get connected again. To assure a possibility to always get connected, any City must use a default register setting during the first 200 milliseconds at start-up.

The following start-up procedure is prescribed for CAN Kingdom Cities:

1 The Mayor performs internal tests.
2 The Mayor puts the Post Office into a bit rate of 125 kbit/s and Silent Mode (no acknowledgment bits nor error frames) for 200 ms waiting for a Default

Letter with Envelope 2031_{Std} containing a Page with all eight Lines containing AA_{hex}.

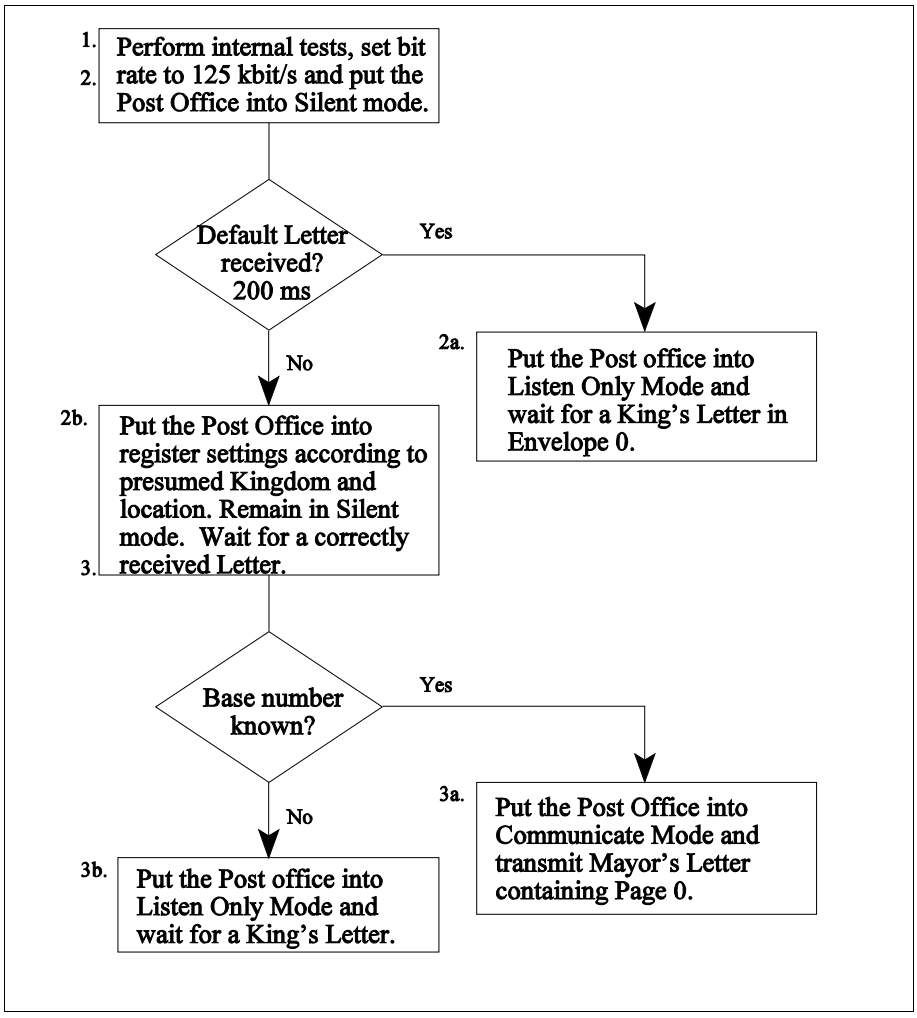
2a If such a Letter is received during the 200ms period, the Mayor keeps the time register settings and switches to Listen Only Mode waiting for a King's Letter.

2b If no Default Letter is received during the 200ms period, the Mayor puts the Post Office into register settings according to presumed Kingdom and location, still in Silent Mode, waiting for a correctly received Letter.

When a Letter is correctly received:

3a If the Base Number is known, the Mayor change the Post Office into Communication Mode and transmits his Mayor's Letter with Page 0.

3b If the Base Number is **not** known, the Mayor switches to Listen Only Mode and waits for a King's Letter.



The difference between Silent Mode and Listen Only Mode is that in Silent Mode, the Post office is completely silent but in Listen Only Mode it participates in message acknowledgment and error handling. When the first City turns into Listen Only Mode, then the King knows that at least one City is attached to the system and alive.

Membership agreement

A Kingdom Founder (system designer) can only take responsibility for his system if he has approved all Cities. No other module should be connected. He has two ways to check the members: By inquiring their identity and by monitoring the bus traffic. Each City in a CanKingdom has an EAN or UPC number. This tells the manufacturer, the product number and the serial number of the City.

The first message the King transmits is the King's Page 1:

Document name:	King's Document
Document List:	0.1 Capital / 0.0 City
Document Number:	0.1 Capital / 0.0 City
Document type:	Transmit (Capital) Receive (City)

Page description.

Page number: 1

Number of Lines: 8

Data description: The King's Page 1. Initiating Page. Provides the Base Number and asks for Mayor's response.

Line description.

The data type of all lines is **Binary.**

Line 0:	City or Group address	
Line 1:	00000001	(Page 1)
Line 2:	xxxxxxxx	The Mayor's respond Page in Mayor's Document.
	11111111	No respond requested.
Line 3:	rrrrrrrr r = 0	Reserved
Line 4:	xxxxxxxx	Base Number, LSB
Line 5:	xxxxxxxx	Base Number
Line 6:	xxxxxxxx	Base Number
Line 7:	Errxxxxx	Base Number, MSB
		All x = 0 -> Keep current setting.
		All x = 1 -> Base Number undefined.
	rr = 00	
	E = 1/0	Extended / Std. format.

When a Mayor receives this Page he will, as soon as possible, respond with the requested Page in his Mayor's Document. Page 0 contains the EAN/UPC code and Page 1 the serial number.

Form for the Mayor's Page 0:

Document name: Mayor's Document.		
Document List: 0.1		
Document Number: 1.1		
Document type: Transmit		
Page description.		
Page number: 0		
Number of Lines: 8		
Data description: City Identification, EAN-13 Code.		
Line description.		
Line 0:	rrrrrrrr	Reserved r = 0
Line 1:	00000000	Page 0
Line 2:	xxxxxxxx	LSB
Line 3:	xxxxxxxx	Product Identification Code
Line 4:	xxxxxxxx	(EAN-13, Check-code omitted)
Line 5:	xxxxxxxx	Unsigned 40-bit integer.
Line 6:	xxxxxxxx	MSB
Line 7:	rrrrrrrr	Reserved

Form for the Mayor's Page 1:

Document name: Mayor's Document.		
Document List: 0.1		
Document Number: 1.1		
Document type: Transmit		
Page description.		
Page number: 1		
Number of Lines: 8		
Data description: City Identification, Serial Number.		
Line description.		
Line 0:	rrrrrrrr	Reserved r = 0
Line 1:	00000001	Page 1
Line 2:	xxxxxxxx	LSB
Line 3:	xxxxxxxx	
Line 4:	xxxxxxxx	Serial Number
Line 5:	xxxxxxxx	Unsigned 40-bit integer.
Line 6:	xxxxxxxx	MSB
Line 7:	rrrrrrrr	Reserved

By using King's Page 1, requesting Mayor's Document Page 0 as reply, each City will reply with its EAN/UPC code. By subtracting the base number from the received Envelope, the King gets the address of each City and by the Page he gets its product number. Then he can check that the right address is associated with the right equipment. A new request for Mayor's

Page 1 will give him the serial numbers and by saving them he can keep track of any replaced City and download system specific settings if needed.

Bus access control

The primary bus access control in CAN is the bitwise arbitration. Even if CanKingdom supports time scheduled or a daisy chained access, the bitwise arbitration is a sound basic mode, allowing for fast unscheduled emergency messages and for the use as a backup mode if the schedule or chain fails. The dependability of a CAN system is to a great extent depending on each message having the right priority. The system designer has full knowledge of the whole system and is responsible for the priority assignment. In a CanKingdom system each City has only one CAN Id at startup, the Envelope for reception of the King's Letter. He gets a transmit Envelope for his Mayor's Page by King's Page 1 and all the rest by Kings's Page 2.

Form for the King's Page 2:

Document name:	King's Document
Document List:	0.1 Capital / 0.0 City
Document Number:	0.1 Capital / 0.0 City
Document type:	Transmit (Capital) Receive (City)

Page description.

Page number: 2
Number of Lines: 8
Data description: The King's Page 2. Assigning an Envelope to or expelling an Envelope from a Folder forming Letters and restraining the use of an Envelope.

The data type of all lines is **Binary.**

Line 0: City or Group address

Line 1: 00000010 (Page 2)

Line 2: xxxxxxxx Envelope Number, LSB
Line 3: xxxxxxxx Envelope Number
Line 4: xxxxxxxx Envelope Number
Line 5: ECrxxxxx Envelope Number, MSB
r = 0 Reserved
C = 1/0 Compressed Envelope yes/no.
E = 1/0 Extended / Std. format.

Line 6: xxxxxxxx Folder Number

Line 7: rrrrrAAD D = 1/0 Enable/Disable the use of this Envelope.

AA = 00 Keep current assignment. (Folder Number is ignored.)

AA = 01 Assign this Envelope to the Folder on Line 6. If another Envelope is already assigned to this Folder, the previous assignment will be kept in parallel. The use of this Envelope is enabled or disabled according to the value of D.

AA = 11 Transfer the current assignment of this Envelope to the Folder on Line 6. The old assignment is canceled. The use of this Envelope is enabled or disabled according to the value of D.

AAD = 100 Expel this Envelope from any assignment. (Folder Number is ignored.)

rrrrr = 00000 Reserved
All other combinations are reserved.

To control the bus access in event triggered systems, it is not enough to have the right priority of each message. It is also necessary to assign a maximum repetition rate of each message as well. This is done by King's Page 10.

Form for the King's Page 10:

Document name: King's Document
Document List: 0.1 Capital / 0.0 City
Document Number: 0.1 Capital / 0.0 City
Document type: Transmit (Capital)
Receive (City)

Page description.

Page number: 10

Number of Lines: 8

Data description: The King's Page 10. Minimum time elapsing between two consecutive transmissions of the same Envelope.

Line description.

The data type of all lines is **Binary.**

Line 0:	City address	
Line 1:	00001011	(Page 10)
Line 2:	xxxxxxxx	Envelope Number, LSB
Line 3:	xxxxxxxx	Envelope Number
Line 4:	xxxxxxxx	Envelope Number
Line 5:	ECrxxxxx	Envelope Number, MSB
	r = 00	Reserved
		C = 1/0 Compressed Envelope yes/no.
		E = 1/0 Extended / Std. format.
Line 6:	ssssssss	Number of segments.
Line 7:	rrrrrrrr	Number of revolutions.

Prohibited period between two consecutive transmissions is given by Line 6 and 7 according to the King's Page 11, Circular Time Base Setup Page.

CanKingdom supports a circular time and Line 5 and 6 above reflects its format. We will not go into the details about the CanKingdom time here but just ascertain that the King can set a maximum repetition rate of each Envelope. This does not only solve the "Babbling Idiot" problem but makes it possible to calculate a maximum latency time and a proper setting for any message[1].

A purely event driven communication or a communication relying on unsynchronized local clocks has the disadvantage that there from time to time will be bursts of messages with 100% busload. One way to avoid that and also to check that all Cities are alive is to daisy chain the messages. One message will trigger the transmission of another message. Such behavior can be set up by King's Page 5. By this Page the King can instruct a Mayor to let a Page in one Document react on another Page in another Document. This feature can be used not only for message scheduling but also for application acknowledgment, triggering events in other nodes,

[1] Ken Tindell, Alan Burns, "Guaranteed Message Latencies For Distributed Safety-Critical Hard Real-Time Control Networks." Technical Report YCS229, University of York, Dept. Computer Science, 1994.

etc. The function can be controlled by enabling/disabling the folders and the link can be deleted by removing a Document from the actual Folder.

Form for the King's Page 5:

Document name:	King's Document
Document List:	0.1 Capital / 0.0 City
Document Number:	0.1 Capital / 0.0 City
Document type:	Transmit (Capital) Receive (City)

Page description.

Page number: 5
Number of Lines: 8
Data description: The King's Page 5. Action Page - Reaction Page

The data type of all lines is **Binary.**

Line 0:	City or Group address		
Line 1:	00000101	(Page 5)	
Line 2:	FFFFFFFF	Folder Number	Folder containing the Acting Document to which the Mayor must react.
Line 3:	PPPPPPPP	Form Number	The Form of the Acting Document to which the Mayor must react.
Line 4:	rrrrrrrr	r = 0	Reserved.
Line 5:	ffffffff	Folder Number	Folder containing the Reacting Document by which the Mayor must respond.
Line 6:	pppppppp	Form Number	The Form of the Reacting Document by which the Mayor must respond.
Line 7:	rrrrrrrr	r = 0	Reserved.

It is often said that CAN cannot be used in dependable systems as it is event driven. This is not true. The ISO 11898 standard does only specify the behavior of the CAN Controller. The bitwise arbitration resolves bus collisions in a time predictable way. CAN does not require collisions, just resolves them if they happen. Nothing prohibits anyone from using CAN in a time scheduled system. CanKingdom supports the creation of a globally synchronized clock within a Kingdom and time scheduling of messages. This topic will not be discussed here and interested parties are directed to the CanKingdom spec. that can be downloaded free of charge at http://www.cankingdom.org.

Runtime control

By King's Page 0 the King tells the Mayors that the setup sequence is completed and that they should bring their respective City into work. It is also used during the run phase to freeze the activity of a City or a Group of Cities, stop transmitting any Letters, reset, etc. Thus, the King can control not only the activity of each City but also the communication. This is sometimes

important during emergency conditions. Parts of the Kingdom can be disconnected from the bus (but still active) offering more bandwidth for the remaining part. When a City is in **freeze** mode, its Mayor must be able to handle all implemented types of the King's Letter. A City can have more than one run, freeze and/or reset mode which can be selected with Page 0. The King can address this letter to a single City, a Group of Cities or all the Cities.

Form for the King's Page 0:

Document name:	King's Document
Document List:	0.1 Capital / 0.0 City
Document Number:	0.1 Capital / 0.0 City
Document type:	Transmit (Capital) Receive (City)

Page description.

Page number: 0

Number of Lines: 8

Data description: The King's Page 0. Terminates the setup phase. Orders a Mayor to set his City into a specific working mode, e.g., in a Run or Freeze mode.

Line description.

The data type of all lines is **Binary.**

Line 0: City or Group address

Line 1: 00000000 (Page 0)

Line 2: rrrrrrAA Action Mode

AA = 00	Keep current mode
AA = 01	Run
AA = 10	Freeze
AA = 11	Reset
r = 0	Reserved

Line 3: rrrrrrCC Communication Mode

CC = 00	Keep current mode
CC = 01	Silent
CC = 10	Listen Only
CC = 11	Communicate
r = 0	Reserved

Line 4: MMMMMMMM City Mode

M = 0	Keep current Mode.
M ≠ 0	Modes according to the City specification.

Line 5:	rrrrrrrr	r = 0	Reserved
Line 6:	rrrrrrrr	r = 0	Reserved
Line 7:	rrrrrrrr	r = 0	Reserved

Action Mode: Relates to actual City Mode and Communication Mode. Action Modes have to be defined by the City Founder.

Communication Modes:

Silent: The Postmaster will be silent but still receiving Letters and notices the Mayor when Letters are accepted. The Postmaster will not transmit any acknowledgment bit nor error or overload frames.

Listen Only: The Postmaster will be fully active but the Mayor will send Letters only upon the King's request.

Communicate: Normal communication.

City Mode: City specific modes, e.g., configuration mode, service mode, working mode, etc. For each City Mode the City Founder has to define how the City will work on Action Mode and Communication Mode commands from the King.

Although the King's Page 0 is the main tool for the King to control his Kingdom during runtime, he is free to use any other Page. Cities can be added or removed during runtime, provided the Kingdom is designed for such behavior. If the Base Number is known a priori, the Mayor will announce himself as soon as he has seen a valid message on the bus, otherwise he will wait to be polled by the King. Then it is up to the King to decide when and how he will be integrated into the Kingdom.

Conclusion

CanKingdom is tailored for designing dependable systems based on the CAN protocol. Due to the separation of system and module design, the system designer is in full control and can take full responsibility for the final system. The system can be reconfigured by control messages. This feature can be utilized not only for a graceful degradation of the system when failures happen, but also for upgrading systems during development or after delivery. CAN systems are often accused of being inherently event triggered and non-deterministic. This is not true. Fully deterministic systems can be designed by CanKingdom, even time scheduled ones. Common problems like membership agreement, bus access and runtime control can be solved by applying CanKingdom.