

---

**CERN**

---

**DIP usage recommendations  
EN-ICE**

**Version 1.1**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## Revision History

Date	Version	Description	Author
20 February 2008	0.1	Initial revision	Mathias Dutour
7 March 2008	0.2	Implemented Wayne Salter comments.	Mathias Dutour
25 April 2008	1.0	Implemented Renaud Barillere comments.	Mathias Dutour
14 December 2011	1.1	Fixed various code samples, updated web links	Brice COpy

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## Table of Contents

1.	Introduction	5
1.1	Purpose	5
1.2	Intended Audience	5
1.3	Acronyms and definitions	5
1.3.1	Acronyms	5
1.3.2	DIP definitions	5
1.4	References	5
2.	Understanding DIP basics	6
2.1	Information over DIP	6
2.2	DIP applicability	6
2.3	Basic DIP principles	6
2.4	Specific DIP characteristics	7
2.5	DIP dynamics	8
3.	Managing DIP deployment and communication	9
3.1	Cross-domains DIP communication on the TN	9
3.2	Special case: WinCC OA (PVSS) as a filtering mechanism	10
3.3	Additional recommendations on Publishers, Subscribers, Publications and DNS	12
3.3.1	Publishers and Subscribers	12
3.3.2	Publications	12
3.3.3	DNS	14
3.4	Technical information concerning deployment	14
4.	Using the DIP API	15
4.1	The big picture	15
4.2	DIP Factory	16
4.3	Subscribers	17
4.4	Publishers	18
4.5	Publications	19

## Table of Figures

Figure 1: DIP dynamics.	8
Figure 3: Cross-domain DIP communication on the TN.	9
Figure 4: Special case: WinCC OA as a filtering mechanism.	10
Figure 5	<b>Error! Bookmark not defined.</b>
Figure 6: DIP packages interfaces.	15
Figure 7: DIP packages dependencies.	15
Figure 8: DIP Factory involved classes.	16
Figure 9: Subscribers involved classes.	17
Figure 10: Publishers involved classes.	18
Figure 11: DIP Publications main classes.	19

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## Table of Recommendations

Recommendation 1: Use the central DNS for any cross-domain DIP data exchanges.	9
Recommendation 2: Use the central DNS whenever possible, including for intra-domain DIP data exchange.	9
Recommendation 3: Identify your domain-private and cross-domains DIP data.	9
Recommendation 4: As a Publisher of cross-domain data, maintain up to date the information over your Publications available for Subscribers.	10
Recommendation 5: Do not regroup Publishers for internal domain data with Publishers for cross-domain on the same machine(s) to simplify communication management.	10
Recommendation 6: Limit the Number of Publishers and Subscribers per domain to achieve good DIP manageability.	12
Recommendation 7: Make sure each Publisher and Subscriber is assigned only one clear (high level) responsibility.	12
Recommendation 8: DIP should not be used as a mechanism to send critical status or operational commands to your systems.	12
Recommendation 9: Apply a consistent logic to structure the Publications offered by the Publishers.	12
Recommendation 10: Apply a consistent logic in the Publications intrinsic characteristics (names, internal structure and refresh rates).	12
Recommendation 11: Adjust the granularity of the Publications' content for their efficient usage on the Subscribers' side.	12
Recommendation 12: For uniformity, preferably use American English for publications' names and descriptions,	13
Recommendation 13: Make use of standard DIP characters and for uniformity, prefer the "/" delimiter for the Publications' names.	13
Recommendation 14: Follow the DIP Publications' naming convention described in [Ref 1].	13
Recommendation 15: Describe accurately the DIP Publications static and dynamic characteristics.	13
Recommendation 16: Make sure the DIP Publications characteristics are maintained up to date and available at all times.	13
Recommendation 17: Communicate any foreseen changes on the DIP Publications for smooth transitions.	13
Recommendation 18: Private DNS(s) shall not be visible outside their hosting domain.	14
Recommendation 19: Make use of the provided DIP test tools and Browser to detect and diagnose communication issues.	14
Recommendation 20: Use unique names when spawning the "DipFactory" singleton.	16
Recommendation 21: Subscribers shall implement and process adequately the entire "DipSubscriptionListener" interface's methods.	17
Recommendation 22: Publishers process adequately the messages received through the "DipPublicationErrorHandler" callback.	18
Recommendation 23: Make proper use of the DIP quality levels on Publisher and Subscribers' side.	20
Recommendation 24: Publishers shall provide a quality reason for changing their Publications' quality to "Bad" or "Uncertain".	20
Recommendation 25: Publishers shall provide a correct timestamp for each of their Publications' update.	20

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## 1. Introduction

### 1.1 Purpose

This document comes in the context of the role of DIP for the LHC operations, and is meant to help users to manage, deploy and configure DIP to match their needs. It addresses the current situation with the DNS located on the General Purpose Network (GPN), as well as when the DNS will be located on the Technical Network (TN). Several topics are covered here ranging from DIP basic understanding, to management and organization of the DIP data distribution and DIP API usage. High and low –level recommendations are provided in various situations.

### 1.2 Intended Audience

This document is meant for any person dealing with data distribution via DIP. It addresses the concerns of team managers, network engineers, developers and integrators.

### 1.3 Acronyms and definitions

#### 1.3.1 Acronyms

**API** = Application Programming Interface  
**DCS** = Detector Control System  
**DIP** = Data Interchange Protocol  
**DIM** = Distributed Information Management  
**DNS** = DIP Name Server  
**DSS** = Detector Safety System  
**EN** = Experimental Network  
**GPN** = General Purpose Network  
**EN-ICE** = EN Department, Industrial Controls Group  
**PVSS** = ProzessVisualisierungs- und SteuerungsSystem  
**WINCC OA** = WinCC Open Architecture (previously known as PVSS)  
**SCADA** = Supervisory Control And Data Acquisition  
**SLS** = Service Level Status  
**TN** = Technical Network  
**UML** = Unified Modeling Language

#### 1.3.2 DIP definitions

For easy identification, every occurrence of a name depicted in these definitions will start with an uppercase letter in the rest of the document.

**Publisher** = A producer of DIP data, independently of software program and applications considerations. A Publisher is responsible for the definition of the structure, the content and the provision of the DIP data to its Subscribers.

**Subscriber** = A client of DIP data, independently of software program and applications considerations.

**Publication** = A logical container that structures related atomic pieces of DIP data. This logic container has a name, as well as timestamp and quality properties that describe its state. This Publication gets its content updated at will by its producer.

**Subscription** = A Subscription refers to a Subscriber’s interest in receiving updates of a certain Publication.

### 1.4 References

- Ref 1:** “LHC Data Interchange Protocol (DIP) Definition”, 2004, Wayne Salter on behalf of the LDIWG, EDMS Reference: 457113  
**Ref 2:** “DIP Name servers: Principles and mechanics”, 2006, Mathias Dutour.  
**Ref 3:** “DIM”, <http://cern.ch/dim>, Clara Gaspar.

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## Understanding DIP basics

### 1.5 Information over DIP

Besides this document and recommendations, there are other sources of relevant information addressing various DIP viewpoints;

One could mention:

- The DIP web pages that presents the DIP news and the available software to download: <http://cern.ch/dip>
- For any DIP questions one could contact DIP experts via our the support email: [icecontrols.support@cern.ch](mailto:icecontrols.support@cern.ch)
- The EN-ICE Piquet is available 24/7 to investigate DNS related issues outside working hours on CERN number **164930**

### 1.6 DIP applicability

As mentioned in [Ref 1] DIP can be described as *“DIP is a system which allows relatively small amounts of real-time data to be exchanged between very loosely coupled heterogeneous systems. These systems do not need very low latency. The data is assumed to be mostly summarized data rather than low-level parameters from the individual systems, i.e. cooling plant status rather than the opening level of a particular valve.”*

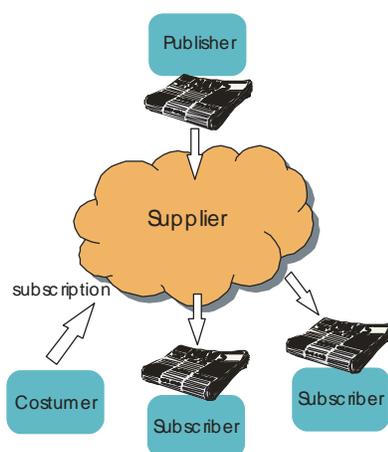
In particular, DIP is a recommended candidate to deal with the following situations:

- Where there is a need for a simple integration of a data distribution mechanism. (DIP doesn't support the transport of services).
- When there is a need to carry high level information between large heterogeneous systems. (E.g. between several controls systems).
- When this information is valuable to multiple systems at once. This is also a reason why DIP shall be used to carry high level information rather than dedicated information. It has also an influence on the structuring of the data propagated through DIP.
- DIP is particularly designed to carry live data or near real-time data.
- When the data conveyed isn't critical, i.e. it doesn't imply security or safety issues.

Beside the plain DIP libraries that are freely available for Windows and Linux platforms with C++ and Java APIs, there are DIP plug-in components being developed by CERN for both LabView and WINCC OA (PVSS).

### 1.7 Basic DIP principles

In one sentence for the techies: “a multicast data distribution mechanism using the push model”.



decided by the publisher.

DIP is essentially an information distribution service, and as such may be compared with subscribing to news papers or magazines:

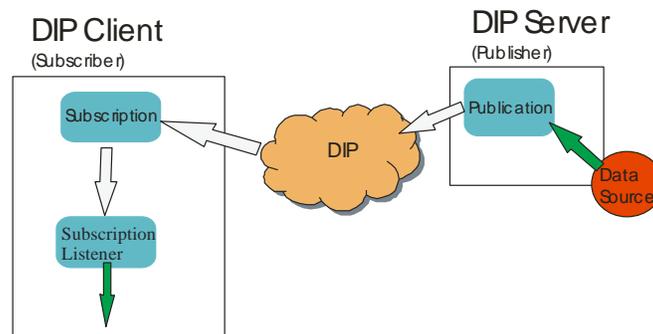
If one wants to receive a magazine, he or she may subscribe to that magazine by giving the name of the magazine to the supplier. Whenever, a new edition of that magazine is published that person will receive a copy of it.

The person is of course not restricted to subscribing to one magazine, or just to magazines from a single publisher. The person need only provide to the supplier names of the publications he or she is interested in (they need not know who or where the publisher actually is) and they will receive new editions of the requested publication as they become available.

DIP is essentially playing the role of the Supplier in the above scenario. There is one notable difference between DIP and the magazine scenario described above. That is a magazine is published on a periodic basis, this is not necessarily the case with DIP publications which are published at a time

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

Important components in the DIP architecture are Publishers, Subscribers and Publications as shown in the diagram below.



See the chapter 3 "Using the DIP API" for more details about Publishers, Subscriber and Publications from a DIP API point of view.

## 1.8 Specific DIP characteristics

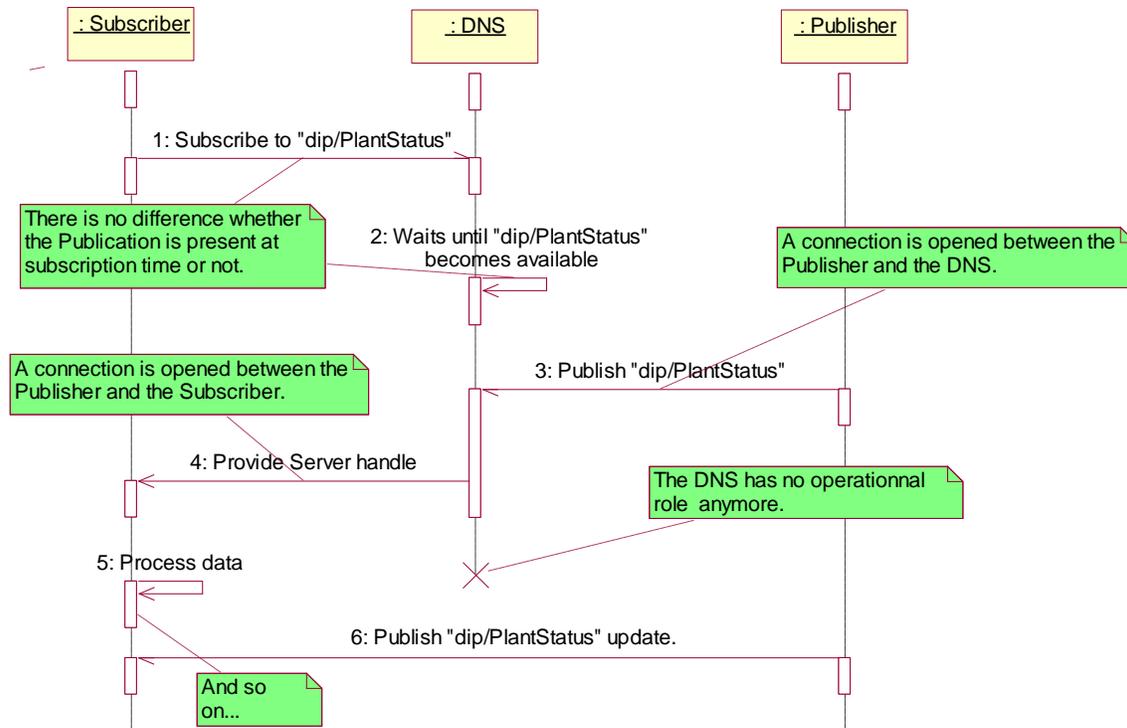
There are some important characteristics about DIP one should be aware of:

- The Publisher and its Subscribers don't know each other explicitly. The DIP protocol, actually the DNS, connects the Subscribers with their Publishers of interest. A subscribers knows at all time the status of its connections to Publishers.
- A Publication is a structured container of atomic data. It is a consistent item and the Subscribers cannot subscribe only to a subset of its content.
- There is no filtering mechanism provided by DIP itself, i.e. one subscribed to a Publication, it will receive all its updates.
- The Publisher has full control over data content, quality and timestamp,
- There is a notion of "data contract" between the Publishers and their Subscribers. A Publisher responsible for a set of Publications is not allowed to change online the structure of its Publications once they have been made available for potential Subscribers.
- There are no particular security mechanisms implemented in the DIP protocol. For example, a Subscriber has no means to authenticate the source of the data it receives. Similarly, the DIP protocol doesn't offer a Publisher the possibility to restrict access to a set of authenticated Subscribers. Another example, the Publications names are not nominative. In other words, when a Publisher stops, the Publications names it was using are from then on freely available to other Publishers.
- There is no feedback to a Publisher that the data published actually reached its Subscribers (also known as "one-way" communication). Hence there is no retransmission in case of a data delivery issue.

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## 1.9 DIP dynamics

The pseudo UML schema below describes the sequence of actions between a Subscriber and a Publisher.



**Figure 1: DIP dynamics.**

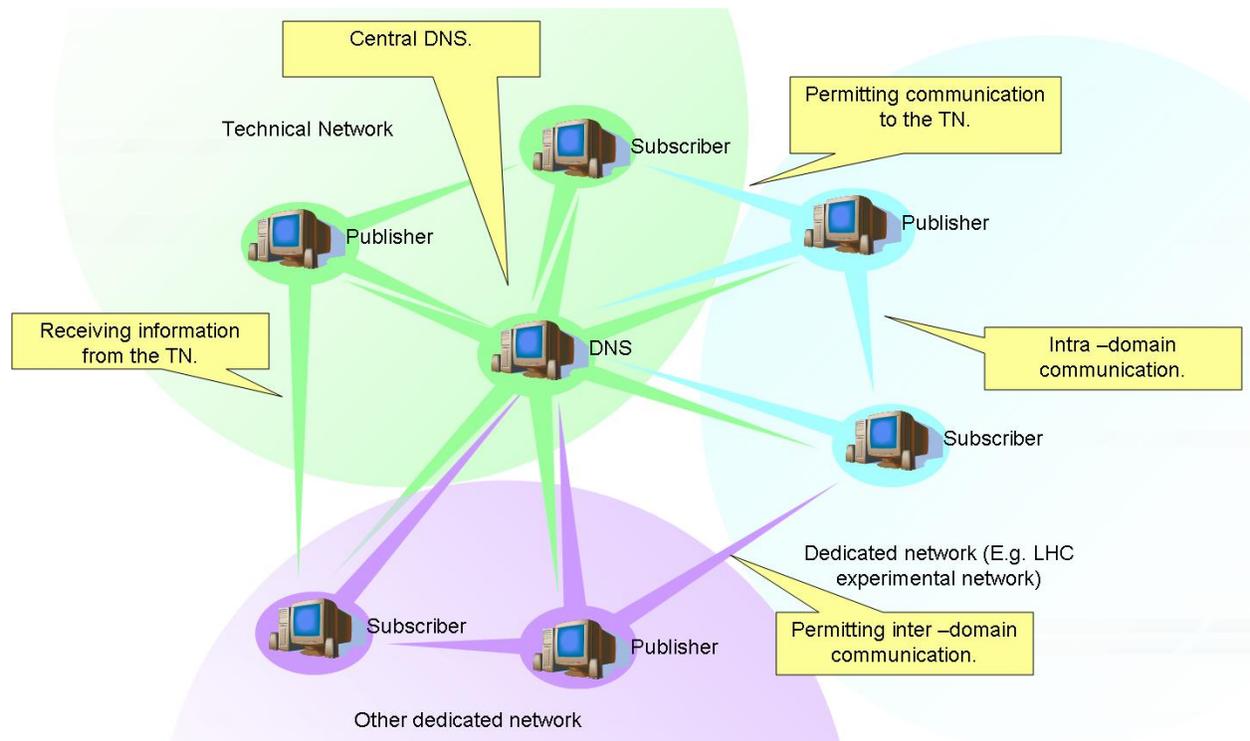
The picture above provides a simplified picture of what's happening when one subscribes to some Publications (here "dip/PlantStatus") as well as the role of the DNS in connecting the Publishers with their Subscribers. More detailed explanations can be found in the DIP FAQ: <http://cern.ch/en-ice/DIP+FAQ>. From this picture, it is easy to understand that the network settings shall be configured adequately to allow the correct communication. These aspects are covered in the next chapter.

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## 2. Managing DIP deployment and communication

### 2.1 Cross-domains DIP communication on the TN

The picture below shows the logical data flows between DNS, Publishers and Subscribers on the TN and on the ENs:



**Figure 2: Cross-domain DIP communication on the TN.**

This picture represent the future DIP organization to be deployed mid 2008, with a central DNS on the TN and various Publishers and Subscribers communicating together across domains. This DNS is maintained by ITCO and therefore will require no maintenance for the experiments.

**Recommendation 1: Use the central DNS for any cross-domain DIP data exchanges.**

**Recommendation 2: Use the central DNS whenever possible, including for intra-domain DIP data exchange.**

Within a certain domain, the DIP communication is typically not constrained and DIP data can be freely exchanged between Publishers and Subscribers. In case there is a need to restrict some DIP data access within a certain domain to a set of Subscribers, this can be achieved by network configuration, filtering on IP addresses. However, this filtering is not so flexible and supposes an “all or nothing” filtering logic on the Publications.

Then, the picture above clearly indicates the connections across domains must be clearly identified and managed. This is achieved by different means:

- First it is important to identify properly which DIP data is of interest to other domains to prevent to mix domain-private and cross-domain information within the same Publications.

**Recommendation 3: Identify your domain-private and cross-domains DIP data.**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

- It is crucial for the Publishers to present the data available for Subscribers and to maintain this information up to date. Once a Subscriber is interested in DIP data being published in another domain, its responsible could contact the Publisher's responsible and together decide to setup a proper network configuration enabling the DIP communication.

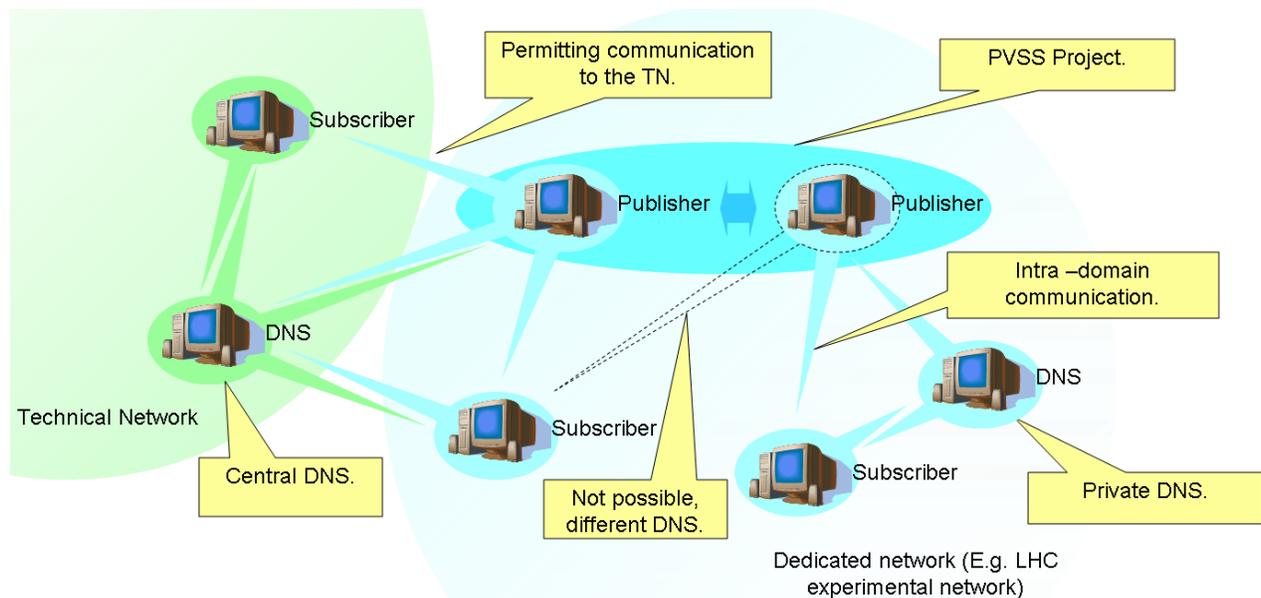
**Recommendation 4: As a Publisher of cross –domain data, maintain up to date the information over your Publications available for Subscribers.**

- Finally, these Publishers providing data for other domains could be collocated on the same machine(s) to simplify communication management (i.e. network configuration.). Furthermore, Publishers of intra domain data shall preferably not be collocated with Publishers for cross-domain data.

**Recommendation 5: Do not regroup Publishers for internal domain data with Publishers for cross–domain on the same machine(s) to simplify communication management.**

## 2.2 Special case: WinCC OA (PVSS) as a filtering mechanism

The picture below describes how one could use a WinCC OA system and a private DNS as a DIP data filter:



**Figure 3: Special case: WinCC OA as a filtering mechanism.**

In this configuration, a domain–private DNS is used to manage all the DIP data that shall remain within the domain while the central DNS is used to expose and exchange DIP data with other domains. Then the WINCC OA (PVSS) internal database is used as a “bridge”, here with a Publisher connected to the central DNS another connected to the private DNS. One could use such mechanism when only a subset of the total data available inside the domain, shall be exposed across domains, possibly structured differently into Publications.

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

There are several advantages to this technique:

- It is simple to setup in a WINCC OA (PVSS) project (only using DIP v.5.2.0 and above).
- It is a flexible mechanism to adjust visibility of Publications.

...And disadvantages:

- There is a maintenance effort required to guaranty the availability of the private DNS in the domain.
- It could become difficult to maintain a clear picture of the Publications' visibility in case there are many Publishers using this "bridge" mechanism in WINCC OA (PVSS).

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

## 2.3 Additional recommendations on Publishers, Subscribers, Publications and DNS

### 2.3.1 Publishers and Subscribers

In order to keep the big picture of your DIP data flows clear and manageable in a domain, it is important not to distribute the DIP data over too many or too few Publishers and Subscribers.

This is achieved by clearly identifying high level responsibilities and then mapping the latter onto Publishers and Subscribers. Often a Publisher is responsible for a “horizontal” technical service such as Cooling and Ventilation, DSS, etc... and therefore is in charge of an homogeneous and logically related set of information. On the other hand, a Subscriber is rather in charge of a “vertical” operational slice such as an experiment’s DCS, relying on many technical services.

So again these responsibilities shall be well scoped to balance flexibility versus complexity.

#### **Recommendation 6: Limit the Number of Publishers and Subscribers per domain to achieve good DIP manageability.**

It is in general advisable to assign each Publisher and Subscriber only one high level responsibility, for example to prevent the unavailability of unrelated Publications in the event of maintenance.

#### **Recommendation 7: Make sure each Publisher and Subscriber is assigned only one clear (high level) responsibility.**

Last but not least, DIP is meant to carry non-sensitive, non-critical data due to the minimum security and data delivery it offers and also due to the lack of visibility on the origin of a Publication. Be aware that when using DIP to send a command, there is no feedback telling whether the command was received, interpreted or ignored. (See 1.8 Specific DIP characteristics).

Hence DIP is either not meant to carry mission-critical commands like “Turn on system X” but may be used with caution for supporting tasks like “Activate logging”.

#### **Recommendation 8: DIP should not be used as a mechanism to send critical status or operational commands to your systems.**

### 2.3.2 Publications

One could apply the previous recommendations to the Publications themselves. Hence it is important to properly apply a consistent logic across your Publication when you define and structure them. This logic shall be natural as it defines the way the Subscribers are going to access this data. Often this Publication’s organization is driven by the logical structure of a system, which shall make sense for the Subscribers, and less often by the hardware architecture or other considerations.

Finally, adjusting the granularity of the Publication’s content will help to maximize usability and extensibility, avoiding the “super” Publication with dozens of data fields just like the atomic one containing little valuable information by itself.

#### **Recommendation 9: Apply a consistent logic to structure the Publications offered by the Publishers.**

#### **Recommendation 10: Apply a consistent logic in the Publications intrinsic characteristics (names, internal structure and refresh rates).**

#### **Recommendation 11: Adjust the granularity of the Publications’ content for their efficient usage on the Subscribers’ side.**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

Each Publication's name must be unique per DNS instance, hence a naming convention was defined in [Ref 1] and summarized below:

“dip”/Domain/[SubDomain]/[SubSubDomain1]/[SubSubDomainx]/Item E.g.,  
dip/ATLAS/Muon/EndCap/Rack/Rack53/Temperature.

Finally, please use the standard characters

(abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789\_) and delimiters (/=(.)) for the Publications' names, as one couldn't foresee their Subscribers' usage.

**Recommendation 12: For uniformity, preferably use American English for publications' names and descriptions,**

**Recommendation 13: Make use of standard DIP characters and for uniformity, prefer the “/” delimiter for the Publications' names.**

**Recommendation 14: Follow the DIP Publications' naming convention described in [Ref 1].**

Then it is important to communicate efficiently DIP detailed information to respect the DIP “data contract” between the Publishers and the Subscribers. Starting from the organization of the Publications themselves, then other Publication's level details shall be provided:

- The description of the Publication's purpose;
- The description of every field of the Publication, together with the unit intended for each field;
- How the Publication is refreshed (cyclically, periodically, manually, on a particular event, etc.) and possibly how often;
- How the timestamp of the Publication is affected by the Publisher (e.g.: when reading the data from equipment, arbitrarily, etc.)
- How the DIP quality of the data is assigned to the Publication.
- Any other useful information, such as when Publications are known not to be available (maintenance period), etc.

**Recommendation 15: Describe accurately the DIP Publications static and dynamic characteristics.**

Of course this information is primarily addressed to the Subscribers, but also relevant to maintain consistency on the Publisher's side, for example when extensions have to be implemented (e.g.: Integrating new equipment leading to new Publications.). So it shall be maintained up to date by each domain and available at all time, preferably in one place only (E.g. web page, interface definition document (IDD), etc.). It is equally important to communicate any changes foreseen that could affect the Publications in place, to guarantee smooth transitions and avoid clashes.

**Recommendation 16: Make sure the DIP Publications characteristics are maintained up to date and available at all times.**

**Recommendation 17: Communicate any foreseen changes on the DIP Publications for smooth transitions.**

Finally, occasionally one asks whether there should be dedicated Publications playing the role of “watchdog” or “heartbeat” to show the health of Publishers in WINCC OA (PVSS). The point is often raised when dealing with very slow updating Publications, raising some doubts on the Subscribers' side whether the Publisher is alive or not. Although there isn't any constraint there, this is in principle not necessary for 2 reasons.

- First each Publication has a quality that indicates whether one can trust or not the content of the Publication.
- And second, the connection's status with a Publisher is known via some dedicated callbacks on the Subscriber's side.

Now there is a situation where a watchdog could be useful. One could for instance use a Publication as a watchdog when the data produced by a Publisher comes from another system (e.g. a physical device), which could be blocked

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

itself and therefore not updating its own data anymore. In this case a watchdog could be implemented between the Publisher and this system, and then published via DIP for the Subscribers, (I.e. to assert the health of the device not of the Publisher).

### 2.3.3 DNS

The role of the DNS is to maintain the list of Publications available, and connect the Subscribers to the Publishers. It is therefore necessary at the time this connection must be established.

The “hot” DNS managed by ITCO is backed-up by a “standby” one in a failover fashion, and the guarantee is granted that the switch is transparent for the Publishers and Subscribers. More details of this mechanism are provided in [Ref 2].

Although there aren’t official scalability performance figures at this time, no particular issues have been noticed, for example when switching more than 20k Publications from the “hot” to the “standby” DNS.

To setup a private DNS, one has to use the DNS provided in the DIM delivery, see [Ref 3]. One shall make sure it is deployed on a machine only visible within the private domain to prevent any risks of duplications and confusion concerning cross-domain Publications managed by the centrally managed DIP name servers.

**Recommendation 18: Private DNS(s) shall not be visible outside their hosting domain.**

## 2.4 Technical information concerning deployment

Deploying DIP applications is rather simple; one has to make sure that:

- Either the centrally managed DIP name server is visible to Publishers and Subscribers, in which case there are no particular settings to be adjusted;
- Or that the domain-private DIP name server or servers are visible.

In the second situation, the DNS “DIM\_DNS\_PORT” environment variable has to be set to 2506. This is the port used by DIP to accept incoming connections.

On the Publishers and Subscribers’ side, the environment variable “DIPNS” shall point to the name of the machine where the private DNS is running. The value associated to “DIPNS” can be a comma –separated list of machine names, hence allowing failover DNS to be declared, e.g.: “DIPNS=pcitco161.cern.ch,pcitco162.cern.ch”.

Finally, the DIP Publishers are using the ports in the range 5100 to 6000, so these shall be opened on their hosting machines, e.g. on a Linux machine (iptables):

```
[...]
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 5100:6000 -j ACCEPT
[...]
```

Finally, the DIP delivery is provided with sample programs which are very useful to check your DIP environment health and to use as a starting point for real development. Additionally, the DIP Browser is also an important tool that can connect to any DNS given that the visibility is granted and the DIPNS environment variables properly set. It is therefore often useful to use the DIP Browser to detect and diagnose DIP communications issues.

**Recommendation 19: Make use of the provided DIP test tools and Browser to detect and diagnose communication issues.**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

### 3. Using the DIP API

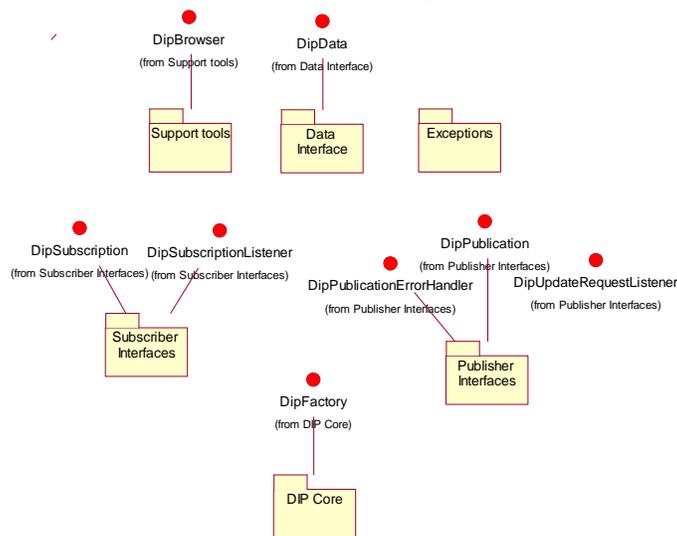
#### 3.1 The big picture

The DIP delivery contains mainly 3 sets:

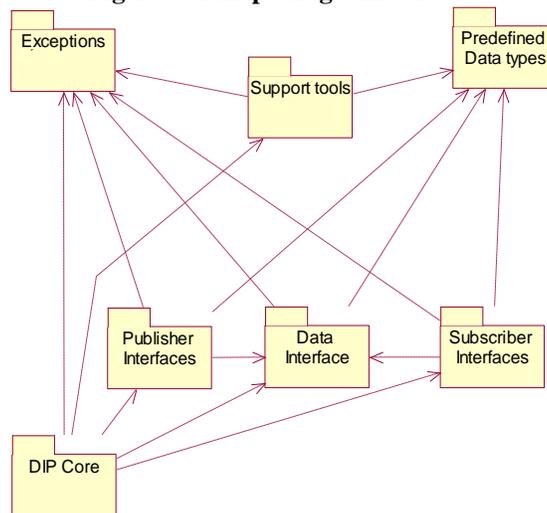
- The DIP libraries (C++ and Java)
- The DIP Browser, a utility tool that connects to the DNS, that actually contains a Publishers that provides the list of Publications it manages
- Sample C++ and Java programs, to give DIP a spin, to present concrete examples and to have a starting point for real development.

Although the packages, interfaces and classes presented in this document originate from the C++ library, the most commonly used so far, all the recommendations apply as well for the Java library.

Here the DIP packages with their interfaces and dependencies are presented:



**Figure 4: DIP packages interfaces.**



**Figure 5: DIP packages dependencies.**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

All UML diagrams of DIP are available in [Error! Reference source not found.]. Each aspect is refined in the following sections.

### 3.2 DIP Factory

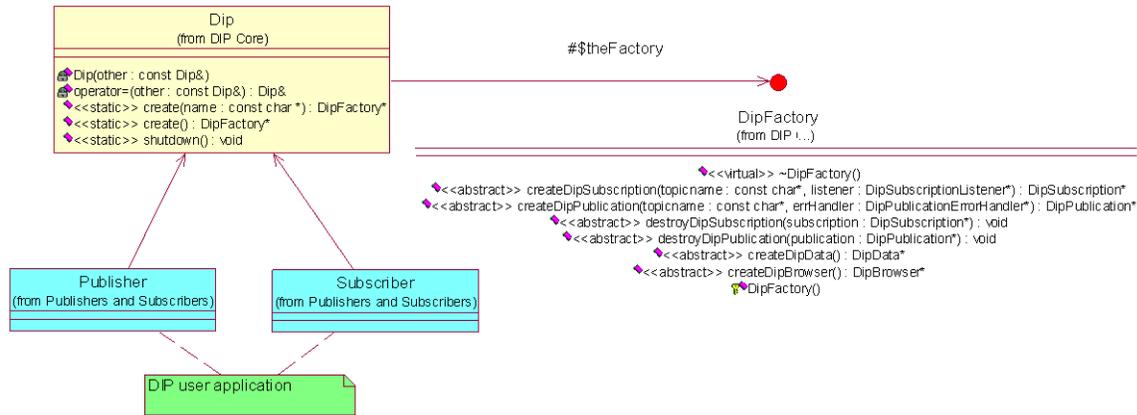


Figure 6: DIP Factory involved classes.

The “Dip” class is an entry point for any Publisher or Subscriber to access the Singleton “DipFactory” class. One important point, it is recommended to use the characterized “create” method of the “Dip” class passing a unique name, as this name will be used to identify the user application handle in the DNS. Hence one could use such construction to guarantee the name is unique when running the user program:

```
// DIP object
DipFactory *dip;DipTimestamp aTimeStamp;
long someRandomness = (long) aTimeStamp.getAsMillis();
string clientName = "MyClientName";
clientName += ltoa(someRandomness);
dip = Dip::create(clientName.c_str());
```

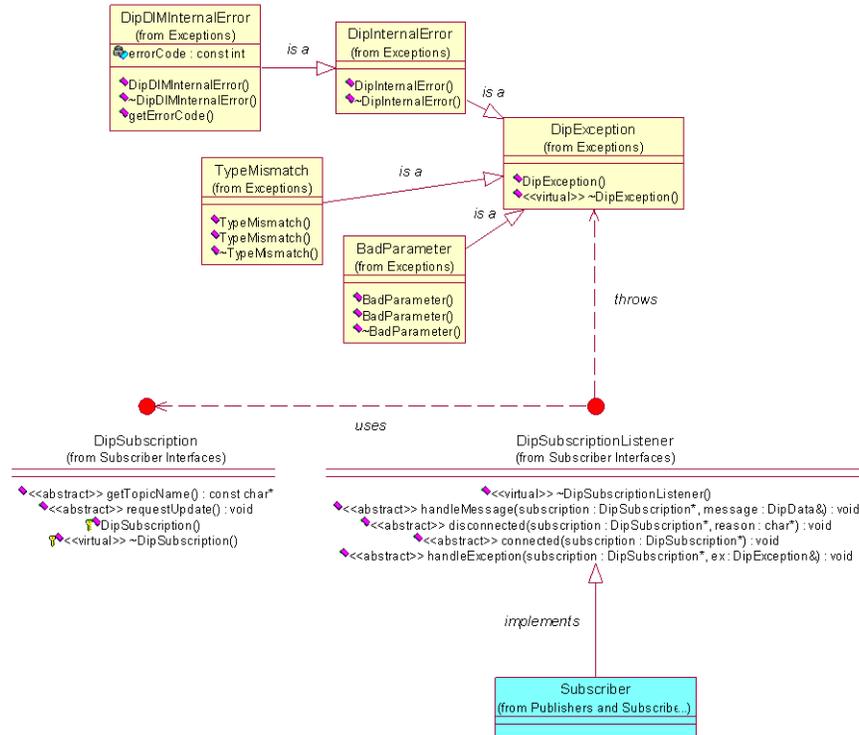
Then, this “DipFactory” class allows the user code to manage the life cycle management of Publications and Subscriptions.

**Recommendation 20:** Use unique names when spawning the “DipFactory” singleton.

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

### 3.3 Subscribers

Here the main classes relevant to the Subscribers are presented:



**Figure 7: Subscribers involved classes.**

The user application implementing the “DipSubscriptionListener” must implement all its methods. In particular the “connected”, “disconnected” and “handleException” methods shall not be neglected as they indicated important information about the health and the status of the connection to the user application’s Publishers.

**Recommendation 21: Subscribers shall implement and process adequately the entire “DipSubscriptionListener” interface’s methods.**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

### 3.4 Publishers

Here the main classes relevant to the Subscribers are presented:

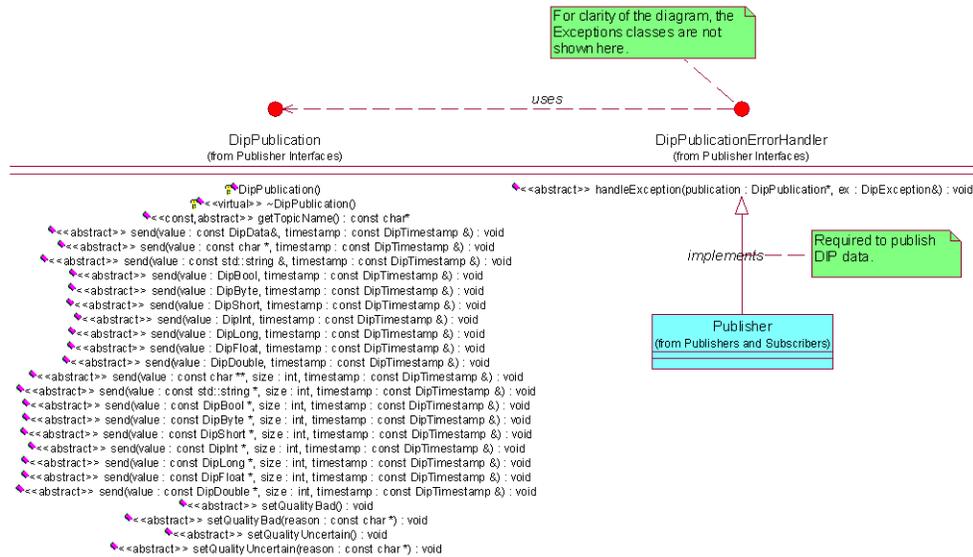


Figure 8: Publishers involved classes.

The Publishers must provide an instance of the “DipPublicationErrorHandler” class and properly handle the potential error messages received.

**Recommendation 22: Publishers process adequately the messages received through the “DipPublicationErrorHandler” callback.**

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

### 3.5 Publications

Here the main classes relevant to the Publications are presented:

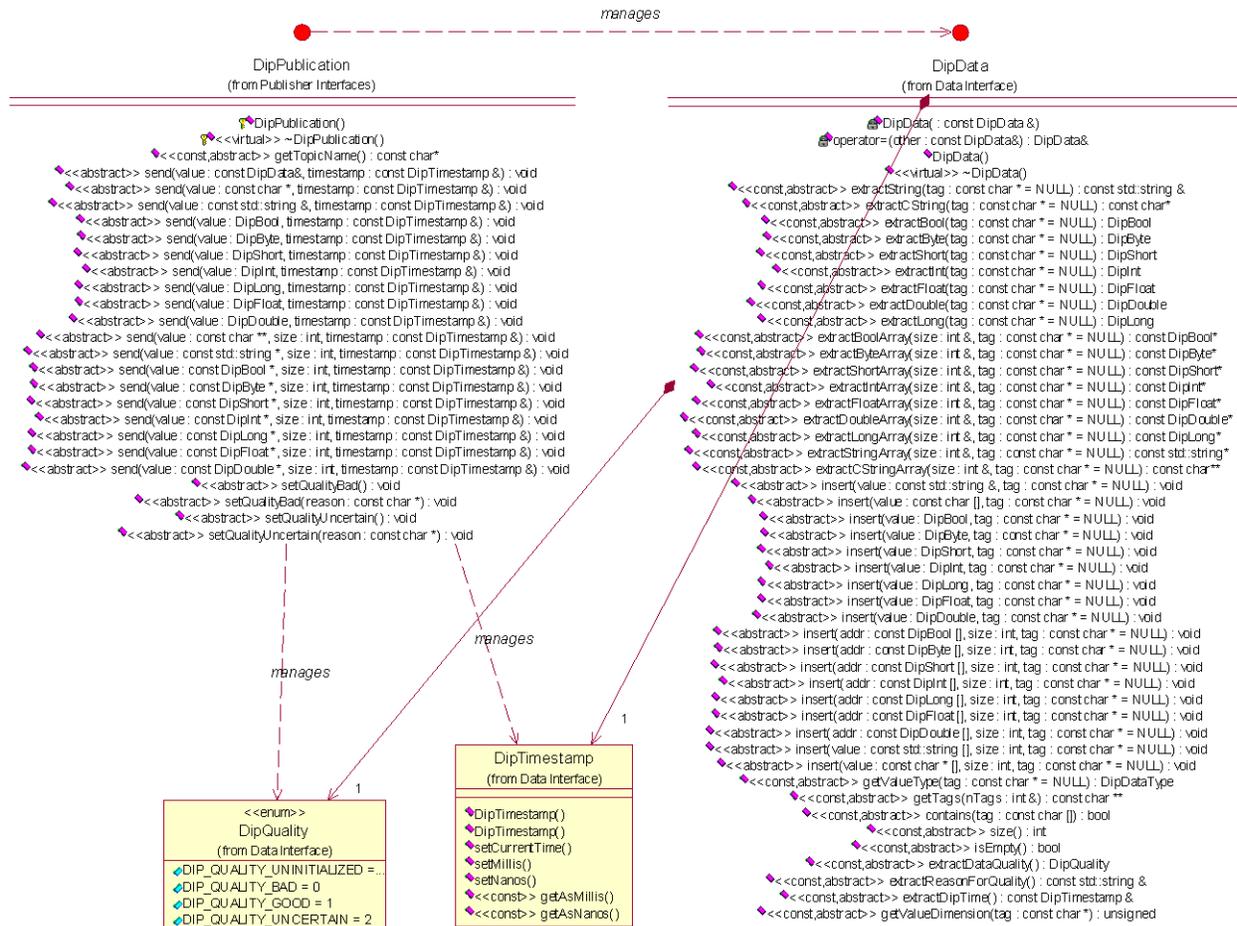


Figure 9: DIP Publications main classes.

As indicated earlier, it is important for the Publishers to assign timestamp, quality and quality reason to DIP Publications correctly and for the Subscribers to understand them.

Although the Subscribers could receive 4 different quality levels, a Publisher can either set the quality of a Publication to “Bad” or “Uncertain”, given that normal updating assigns a good quality and “Uninitialized” is managed by DIP.

Quality	Description
DIP_QUALITY_UNINITIALIZED	Indicates the DIP data object contains uninitialised data.
DIP_QUALITY_BAD	Indicates that data can not be used.
DIP_QUALITY_GOOD	Indicates that data can definitely be trusted. The last update from the publisher was successful and the value(s) that this quality corresponds to are the most current available.
DIP_QUALITY_UNCERTAIN	Indicates that the last update attempt by the publisher had failed (meaning the Publication data source was not accessible). The value(s) this quality corresponds to is the last known good value by the Publisher which can no longer be considered up-to-date.

User manual	Version: 1.1
DIP usage recommendation	Date: 14/December/2011
DIPUsageRecommendations.doc	

**Recommendation 23: Make proper use of the DIP quality levels on Publisher and Subscribers' side.**

Additionally, the Publisher has the possibility via the API to provide for “Bad” and “Uncertain” qualities, a reason for setting these quality levels, always useful to the Subscribers.

**Recommendation 24: Publishers shall provide a quality reason for changing their Publications' quality to “Bad” or “Uncertain”.**

Finally, the timestamp provided with each Publication's update is meant to describe when the data provided was actually measured. This could be different than the time at which this update is sent to potential Subscribers.

**Recommendation 25: Publishers shall provide a correct timestamp for each of their Publications' update.**

END OF DOCUMENT.