SAFARI

__S__calable __A__nd __F__lexible optical __A__rchitecture for __R__econfigurable __I__nfrastructure

Deliverable D3.3

White paper and/or input to ONF/OIF on the abstraction mechanism

Issue date: April 5th, 2016

Author(s):	**Name**	**Partner**
	Ulrich Häbel	Coriant
	Klaus Pulverer	Coriant
	Tetsuro Inui	NTT
	Akira Hirano	NTT
	Yutaka Miyamoto	NTT
Checked by	Toshio Morioka	Technical University of Denmark

EUJ-2-2014: Optical communications
Grant agreement for: Research and Innovation action
Grant agreement no: 642928

Start date: October 1, 2014 Duration: 36 month

Dissemination Level		
PU	Public, fully open, e.g. web	X
CO	Confidential, restricted under conditions set out in Model Grant Agreement	
CI	Classified, information as referred to in Commission Decision 2001/844/EC	

Executive Summary

This white paper explores the use-case and required functionality for scalable and flexible optical transport networks using multi-core fibres (MCFs). Then, it discusses the structure of SDN control information and proposes a set of extensions and flow abstractions in support of OCh circuits across multi-core fibres. It summarizes the current state of optical extensions for OpenFlow-switch protocol and identifies limitations and necessary enhancements. Considering SDM switching concepts evaluated by the ongoing EU project INSPACE it uses the notion of a spectral-spatial flow to describe simultaneous spectral and spatial content switching as an abstraction of a layer 2 packet flow. The document proposes a set of OpenFlow-switch protocol element extensions in support of spectral-spatial flows. Narrowing the focus to the implementation of the control component for the SAFARI testbed it proposes a realization based on the Open Daylight FlowNode data model and an information transport mechanism based on NETCONF.

Table of Contents

1 Introduction

1.1 Scope

The SAFARI project aims to develop programmable optical hardware, and space-division multiplexing (SDM)-based optical component technologies capable of realising highly scalable & flexible optical transport networks for the long term future. This document elaborates on the abstraction mechanism and control function extensions needed to efficiently manage programmable hardware. Based on SDN paradigms and current ONF specifications defining OpenFlow-switch protocol extensions in support of OTN networks [1], it addresses enhancements in support of spatial multiplexing in flexible grid WDM capable transport networks. This document also references SDM switch concepts derived by the ongoing ICT project INSPACE [2]. In preparation for the experimental implementation and showcase of the project the document proposes extensions to the Open Daylight data model leading to a sample SDN application to control programmable hardware.

2 Use-case

2.1 Ultra-high capacity optical transport network

As discussed in Deliverable D3.1, one of the most important use cases of ultra-high capacity optical networks is represented by a recovery scenario from network failures such as catastrophic disaster and disruptive traffic congestion. As the model network for this use case we consider optical ultra-highways between data centres, using MCFs as the transmission line. The network control layer can establish, delete or change ultra-high capacity parallel connections on-demand. In the process, it is required to control numerous parameters governing several tens of nodes, nearly hundred wavelengths, over thirty cores, several modulation formats even in a single ROADM system. In order to manage the resulting complexity, a hierarchical controller model is indispensable for the information abstraction of the numerous parameters in the ultra-high capacity optical transport networks using MCFs.

2.2 MCF deployment scenario

We considered the MCF deployment scenario and clarified the impact of the inter-core XT on network planning and provisioning. Figure 2-1 and 2-2 show the XT issue in the MCF deployment scenario in future ultra-high capacity optical networks. Here, we consider the use case that the MCF is incrementally deployed in multiple steps to realize the future ultra-high capacity optical network. Firstly, the network is composed of conventional single-mode fibres (SMFs). In the second step when we deploy MCF in the highly congested section between A and B where the optical path is short and the XT meets the guaranteed value, it is not necessary to consider the XT value in network design (XT-tolerant). At this stage, the deployment of MCFs is only sparse. However, when we extend the MCF section to site C, if the optical path is long, the XT value is not negligible, it would be necessary to select the applicable modulation format depending on the XT. The non-sparse MCF deployment represents a network that requires careful XT consideration for network planning and operation.

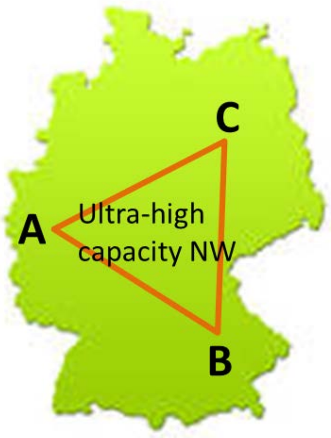

Figure 2-1: An example of future ultra-high capacity network in Germany

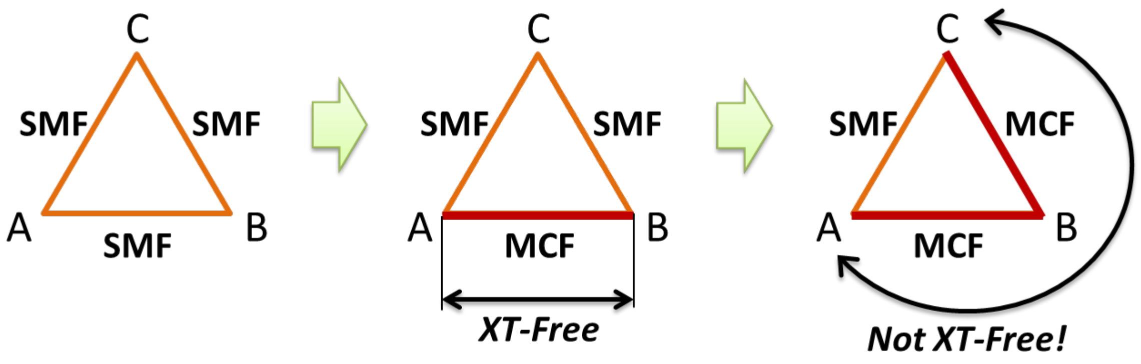

Figure 2-2: MCF deployment scenario

3 Required functionality

3.1 XT related network planning and provisioning

Figure 3-1 shows the XT related network planning & provisioning for programmable ultra-high capacity optical transport networks. The graph shows the Q-penalty induced by XT in case of various modulation formats as a function of the transmission distance. There are two operational modes: conventional operation and emergency operation.

If the distance is short and the XT is negligible (XT-Free), the cores can be bundled. We can treat the link as a multi-core aggregated link and realize the information abstraction, even if a high number of cores per fibre significantly increases hardware and software resource demand when deploying MCFs. We discuss the information abstraction mechanism for the space-division-multiplexing in detail in the following chapters. In the intermediate distance where the XT is not negligible, there will be a restriction on modulation format. Here, we can select applicable modulation formats depending on their XT tolerance.

In emergency operation, network operators may need to provision a long distance detour route for critically important services, whose length is beyond the allowable XT penalty for normal modulation formats. To serve this scenario, restrictive measures have to be supported in order to reduce XT to a manageable level for critically important services. Transmission may be restricted to a sparse distribution of cores in order to reduce inter-core XT. In addition, non-overlapping wavelengths distributions in adjacent cores may be used, and XT monitoring information (see for example [3]) may be taken into account, which is being investigated in WP4. Furthermore, there could be other methods such as using an enhanced forward error correction (FEC) and reducing the number of sub-carriers. A centralized control layer is required to support these functions in order to give network operators continuous controllability for the ultra-high capacity optical transport even in such a disaster recovery use case.

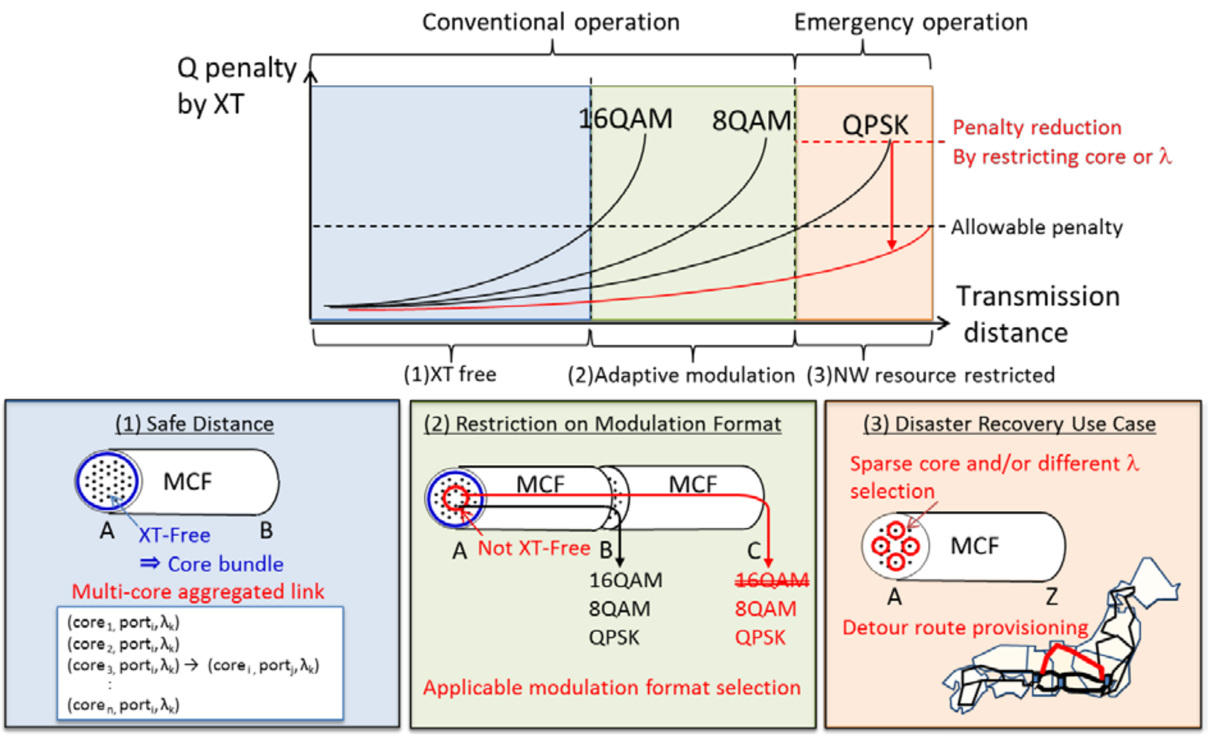

Figure 3-1: XT related network planning and provisioning for ultra-high capacity optical transport

4 Software-defined Networking for optical transport networks

SDN is a network architecture that decouples control and forwarding functions with the goal to achieve directly programmable networks. The concentration of control functions in a centralised SDN controller enables flexible network resource abstraction and virtualisation for business applications. Within this architecture the SDN controller coordinates the forwarding state of network devices through an abstraction called an OpenFlow switch. In addition to network virtualisation, the controller may implement a variety of network services such as bandwidth management, traffic engineering, policy management and others. Business applications have access to these network services by programmable interfaces defined by various internet standards. Core network infrastructure management such as equipment installation and maintenance or software upgrade are not considered in scope for SDN.

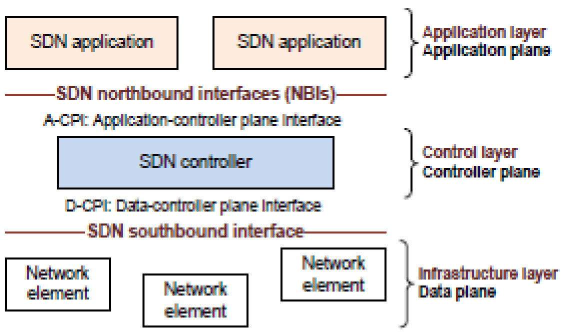

Figure 4-1 Basic SDN components [4]

With an initial focus on layer 2 networks, SDN started to generalize its architecture to expand control to cover L0/1 transport layers networks.

This chapter summarizes the main SDN architecture building blocks and current status of optical extensions with a focus on concepts and constructs that may be extended to support spatial multiplexing.

4.1 OpenFlow Switch

The OpenFlow switch abstraction is composed of one or more flow tables and a group table representing forwarding configuration, and an OpenFlow channel which acts as a control channel to an external controller. Control communication uses the OpenFlow-switch protocol. Physical switch packet processing is modelled as a sequence of match/action operations configured by the external controller where a specific flow is identified by a set of match fields. Flow entries are organized in flow tables. A match field refers to specific packet header such as its source/destination address or tags. In case where an incoming packet matches the set of fields describing a specific flow entry, the instructions associated with that flow are executed. An instruction may route the packet to the next set of flow entries or store an instruction in the packet's ACTION_SET. After having passed flow table processing, the ACTION_SET is executed (in a specified order which is not necessarily the order in which the actions have been added to the ACTION_SET) and the packet is forwarded to the port specified in the output action.

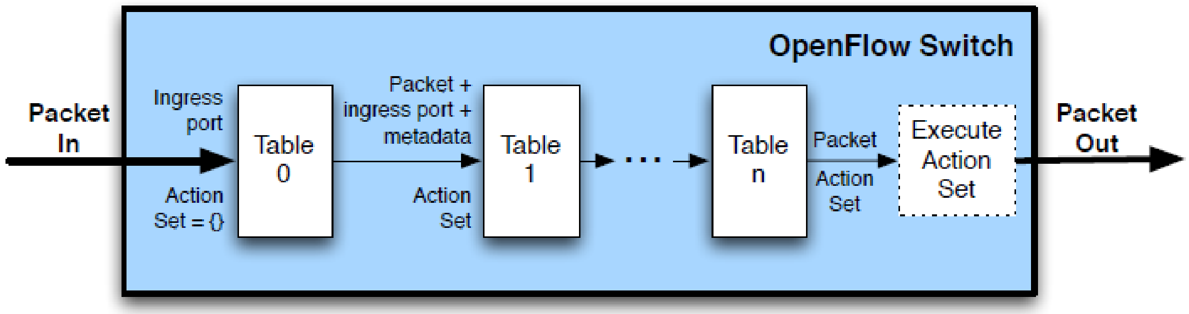

Figure 4-2 Packet flow through processing pipeline ([5])

Each flow table must support a specific table-miss flow entry which defines the forwarding action in case where the packet does not match any flow criteria. Typically, that action may either drop the packet or pass it to the SDN controller (or another table) for further inspection. The SDN controller may then create a new flow entry providing instructions appropriate for this newly installed flow. The packet will be passed back to the switch and processed according to the new flow entry.

Flow match and action configuration is maintained by the external SDN controller utilizing the OpenFlow-switch protocol [5]. The OpenFlow-config protocol is intended to control initial physical switch configuration such as resource partitioning into multiple logical OpenFlow switches or the enabling/disabling of individual ports. While OpenFlow-switch protocol operates at the time-scale of packet flow modifications, OpenFlow-config operates a slower time-scale. The protocol transport mechanism reflects the different application scope: OpenFlow-switch protocol encoding is carried directly on a secure TCP connection (usually TLS) whereas transport of OpenFlow-config [6] is a NETCONF based protocol implementing a remote procedure call paradigm [7].

4.2 Optical transport extensions

Although the original field of application for the OpenFlow switch model started as packet switching, its extension to circuit switching networks started in 2014 when ONF published a set of target use cases and a requirement analysis. ONF bundled activities in that area in the Open Transport (formerly Optical Transport) project which is hosted by the Specification Area in ONF. The Open Transport project is chartered to address SDN and OpenFlow control capabilities for transport technologies of various types including optical and wireless transport [8]. It is expected to identify use cases, define SDN architecture, information modelling and OpenFlow protocol extensions and APIs. This section focusses on the initial outcome of that project which recommends a set of OpenFlow-switch protocol extensions accommodating OTN networks.

The ‘Optical Transport Protocol Extensions’ (OTPE) [1] is a standalone document describing a set of enhancements to the OpenFlow-switch protocol using an Optical Transport ‘experimenter’ scope. Protocol enhancements recommended in that document may be integrated into standard OpenFlow protocol at a later point in time. The document defines the following protocol extensions:

- Match/action in support of OTN connections
- Port attributes in support of OTN interfaces
- Support for transport adjacency discovery

New match fields representing the optical signal are introduced. They either identify the grid type and frequency slot for an OCh connection or the Tributary Port number and Tributary slot for an ODU connection. The outgoing signal type is configured according to the SET_FIELD action which takes the new match field format as input. A FLOW_MOD message instantiates a new switch configuration and represents a cross-connect command when applied to a transport layer switch.

With OpenFlow, the length of each match field is constant. Therefore, the new match fields identify a single wavelength or ODUk signal only. Matching a group of optical signals can be achieved by message bundling.

The document introduces a new port property type (`OFPPDPT_OPTICAL_TRANSPORT`) which references the base signal type of the optical or electrical signal accessible at that port. Signal type specific attributes of that port are encoded as ‘features’. E.g. the optical feature of a port is characterized by its Optical Interface Class (OIC) and standard application code as defined in G.698.1/.2 and G.959.1 and G.695. The encoding format also supports private application codes. Unfortunately, the encoding does not include means for scoping private application codes to a specific context. For electrical signal types, the port feature describes the layer stack terminating the signal.

Optical port properties are retrieved by the SDN controller as part of the switch discovery process. The extension does not define a procedure to instantiate a specific port property such as the application code.

A second port property type (`OFPPDPT_ADJACENCY_DISCOVERY`) is introduced that allows exchange of identity information (SENT, EXPECTED and RECEIVED) supporting adjacency discovery. Technology specific encodings are used to program and monitor the respective Trail Trace Identifiers. In general, the optical extensions leverage concepts and encodings defined by ITU-T and IETF: Signal type encodings reuse GMPLS label formats, port capabilities refer to ITU-T optical interface specifications such as applications codes and TTI overhead information. OAM and connection protection applications as well as optical extensions at the CVNI (Control Virtual Network Interface) to a parent SDN controller are not covered by the initial version of the recommendation and are left for further study.

There are some limitations in the OTPE that need to be relaxed for wavelength- and spatial- multiplex applications:

- Modulation format encoding is tied to standard applications codes (G.695, G.698.1, G.698.2, G.959.1). Private application code scoping is missing.

Following current standards, the OTPE extensions do not consider signal characteristics such as modulation format or forward error correction as independent port attributes. Instead, optical transmission port attributes are summarized in application codes. Only a few codes intended for metro applications are defined by ITU-T today. In general, the extensions permit the use of private application codes but do not indicate their scope. Therefore, any private application code must be globally unique.

- Application codes are not programmable

The new optical port properties are exchanged during interface discovery only. OTPE does not describe a procedure that would permit the instantiation of a specific application code and therefore lacks a method to program connection endpoints with attributes such as modulation format and forward error correction. One option to add this capability would require extension of the PORT_MOD OpenFlow message to instantiate a specific application code. The optical port feature should be extended to carry state information indicating the current active application code.

- Limited support for composite wavelength signals

Optical signal match fields do not support grouping of elementary signal types such as a group of wavelengths/slots. E.g. a bi-directional optical connection cannot be switched by a single FLOW_MOD message. Instead two messages are needed. Since OpenFlow 1.4, the protocol supports message bundling which can be used to combine and process multiple FLOW_MOD messages as one atomic switch operation. However, the number of components of a composite optical signal may be significant and force a serious control flow overhead.

In order to operate wavelength-spatial super channels, it would be necessary to relax this limitation and to support flexible switching of compound signal types as further explained in the following section.

5 SDN control for spatial multiplex enabled optical transport

The Safari project explores the use of multi-core fibres with core count >30 to boost the capacity of optical transport beyond current transmission limits. A high core count significantly increases transmission capacity which in turn requires the development new optical switch architectures that are able to switch traffic at Pbps range at reasonable cost. The ongoing EU project INSPACE [9] is chartered to investigate the network aspects and node design of flexible optical networks and their extension into the spatial switching domain. That project evaluated several switch architectures and investigated switching different combinations of wavelength-spatial signals [2]. This section focuses on the implications of that work for SDN applied as to wavelength-spatial flow switching and termination and proposes a set of extensions for the OpenFlow-switch protocol.

5.1 Matching composite WDM/SDM flows

During transmission, spectral signals carried on different spatial modes within a multi-core, multi-mode or few-mode fibre may influence each other causing spatial mode coupling. The coupling may affect only groups of modes or all modes carried in a single fibre. In order to recover the individual spectral-spatial signal, specific decoupling techniques such as MIMO devices need to be deployed at the connection endpoints. A spectral-spatial flow model should support various groupings of individual spectral-spatial signals that are routed along the same path across the network. In particular, the model should also support a simple description of exhaustive flows, i.e. consisting of all spectral constituents within a single spatial mode or all spatial constituents for a single wavelength. From a switch perspective, this would cover the following applications (Figure 5-1):

a) Independent spectral-spatial switching
b) Switching performed on spatial mode basis across all wavelengths
c) Switching performed on wavelength basis across all spatial modes
d) Switching performed on wavelength basis across spatial mode subgroups

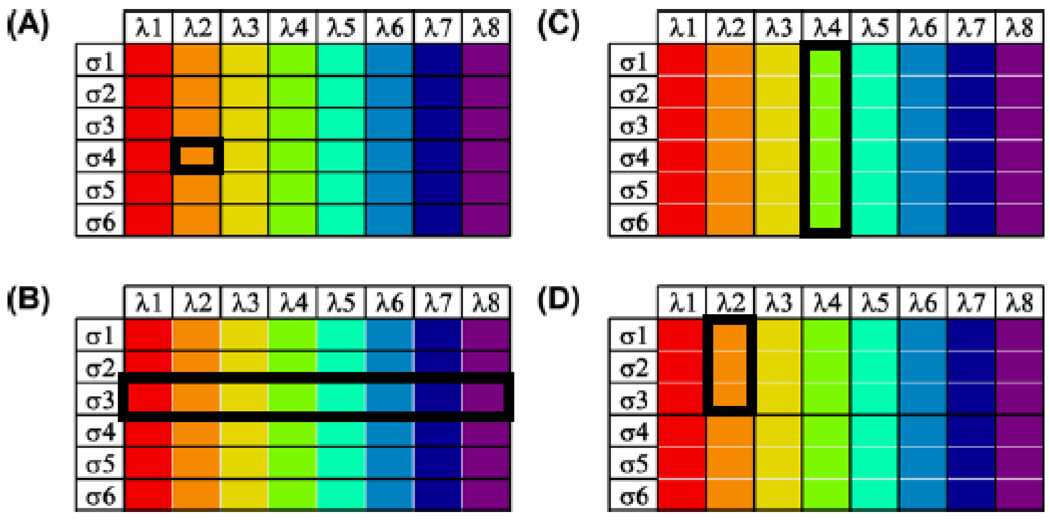

Figure 5-1 (A) Each mode/wavelength channel can be switched independently (B) Switching performed on mode basis across all wavelengths (C) Switching performed on wavelength basis across all modes (D) Switching performed on wavelength basis and spatial mode sub-groups [2].

Beside physical constraints such as spatial mode coupling, flow grouping is also helpful from traffic engineering perspective to reduce the amount of control traffic and computational load for the SDN controller and core network elements. Hierarchical spectral-spatial tunnels push smaller bandwidth flow processing to the border of the core network. This reduces flow control and monitoring overhead not only at the optical core nodes but also at the supervising controller. However, further partitioning of flow control domains seems necessary to manage the large amount of transport connections in spectral-spatial networks.

5.1.1 Composite spectral-spatial flow

Each flow table entry consists of a set of match fields and instructions which is maintained by the SDN controller and issued by a single FLOW_MOD message. The flow table entry matches only if every single constituent match field is satisfied (AND condition). A grouping of flows as described above requires a different evaluation of match fields in such a way that already a match for a single constituent identifies a given flow (e.g. a specific wavelength out of a set of wavelengths).

An elementary spectral-spatial flow can be identified by the tuple:

$$(\text{port}, \text{spatial mode}, \text{spectral component}) = (p, \sigma, \lambda).$$

Port and spatial mode may be combined and represented by a single identifier. The spectral component may refer to a fixed grid wavelength or a flex grid frequency slot.

The above definition can result in rather lengthy lists of elementary flows, which is not very efficient. We propose an optional abbreviated notation to deal with contiguous lists and ranges of flow components. Taking spatial modes as an example:

List notation:

$$\{(p1,\{\sigma1,\sigma2,\sigma3\},\lambda1)\} := \{(p1,\sigma1,\lambda1),\ (p1,\sigma2,\lambda1),\ (p1,\sigma3,\lambda1)\}$$

Range notation:

$$\{(p1,[\sigma1,\sigma2],\lambda1)\} := \{(p1,\sigma,\lambda1)\quad \text{for all } \sigma1<\sigma<\sigma3\}$$

A complex composite flows may be described by decomposing it into a non-overlapping set of contiguous and elementary flows.

An example notation of a composite switch command for a set of spatial modes would be:

$$\{(p1_{in},\sigma1_{in},\lambda1),\ (p1_{in},\sigma2_{in},\lambda1)\} \rightarrow \{(p1_{out},\sigma1_{out},\lambda1),\ (p2_{out},\sigma2_{out},\lambda1)\}$$

Another example shows the switch command for a group of frequency slots and two spatial components

$$\{(p1_{in},\{\sigma1_{in},\sigma2_{in}\},\{fs1,fs2,fs3\})\} \rightarrow \{(p1_{out},\sigma1_{out},\{fs1,fs2,fs3\}), (p2_{out},\sigma2_{out},\{fs1,fs2,fs3\})\}$$

The target spatial modes may be non-contiguous. The notation does also support combination or separation of composite flows. The spectral components are preserved: Wavelength conversion requires OCh signal regeneration which is realized at OCh termination points only.

In order to carry a composite flow switch command in a single FLOW_MOD message it would be necessary to encode lists of match fields identifying a composite flow. This would require OpenFlow to

- Accept variable length match fields
- Consider a table flow entry to match a given flow if at least one match condition is fulfilled

In the case where the composite flow is contiguous (spectral and spatial), the match criteria can be encoded as a set of independent, fixed length spectral-spatial ranges. Only in that special situation, the match fields have a fixed length and comply with current OpenFlow protocol definition.

In order to encode a spatial switch command switching all wavelengths to a different spatial mode, OTPE requires the controller to issue a FLOW_MOD message for each individual wavelength. This would require up to 100 individual FLOW_MOD messages for a single spatial switch command per direction. Combining the information into a single FLOW_MOD message would increase the size of the message but significantly reduce the amount of overhead in total.

In order to reduce the number of FLOW_MOD messages and support composite spectral-spatial signals, new match fields identifying a list of spectral components could be added to the protocol. In addition, combinations of composite flows should be supported to accommodate the aforementioned groupings.

In the following sections we propose a new match type and match fields supporting spectral-spatial flow matching as suggested above.

5.1.2 Conditional-OR match type

OpenFlow currently supports a single match type: The extensible match (`OFPMT_OXM`). A new match type is proposed (OXM_OR) which consists of a variable number of extensible matches. The new match type identifies a specific flow whenever one of the constituent extensible matches is successful. This new match type must not overlap with any other flow entry in the same table. This requires that none of the constituent extensible matches overlaps with any other flow entry.

On the other hand, the set of extensible match criteria can also be realized by a sequence of bundled FLOW_MOD messages where each message encodes a single extensible match. Assuming that a composite flow in most cases can be represented by a small number of extensible matches, the encoding as a bundled set of FLOW_MOD messages may be a reasonable fall back implementation in case the new match type is not supported by the OpenFlow switch.

5.1.3 Composite spectral flow match

This match field describes an ordered list of OCh signals (either by fixed grid centre frequency, or frequency slot boundaries). The construct follows the extensible match field format scoped by experimenter id and makes reuse of OTPE match field definitions. The match construct has a variable length payload which cannot be used to define the composite frequency match field identifier. Instead, the composite spectral flow match field consists of a single 32bit integer storing the total length of the match condition including the list of frequency slot references. Each OCh signal consists of a signal type followed by a signal description match field as defined by OTPE.

```
/* Composite spectral match field.
 * The OXM_HEADER is followed by four bytes encoding the number of individual OCh signals. */

#define                    OXM_EXP_COCH                    OXM_HEADER
(OFPXMC_EXPERIMENTER,OFPXMT_EXP_COCH_SIGTYPE,4)

/* OCh signal type encoding follows the recommendations in Optical Transport Protocol Extensions */

struct ofp_oxm_exp_OCH_sigtype {
        uint32_t oxm_header;      /* oxm_class = OFPXMC_EXPERIMENTER */
        uint32_t experimenter;     /* Experimenter ID which takes the same
                                  form as in struct ofp_experimenter_header. */
        uint8_t sigtype; /* OCH Signal Type */
};

/* OFPXMT_EXP_OCH_SIGID Payload format */
struct ofp_oxm_exp_OCH_sigid {
        uint32_t oxm_header;      /* oxm_class = OFPXMC_EXPERIMENTER */
        uint32_t experimenter;    /* Experimenter ID which takes the same
                                   * form as in struct ofp_experimenter_header. */
        uint8_t grid_type;        /* Grid Type */
        uint8_t chl_spacing;      /* Channel spacing */
        uint16_t n;               /* n is used to calculate the frequency as in
                                   * [ITU G.694.1]
```

```
                                * Frequency(THz)= 193.1 THz + n*chl_spacing (THz)*/
        uint16_t m;            /* m is used to identify the slot width as defined in
                                * [ITU G.694.1],
                                * Slot Width (GHz) = m*12.5 (GHz)
                                * For fix grid networks, m=1 */
};
```

5.1.4 Composite spatial mode match

This match type describes an ordered list of spatial modes. The encoding has a variable payload length and follows the same mechanism as the composite spectral flow match.

```
/* Composite spatial mode match field.
 * The OXM_HEADER is followed by four bytes indicating the number of individual spatial modes. */

#define OXM_EXP_CSM OXM_HEADER (OFPXMC_EXPERIMENTER,OFPXMT_EXP_CSM,4)

/* OFPXMT_EXP_SM_SIGID Payload format */
struct ofp_oxm_exp_SM_sigid {
        uint32_t oxm_header;    /* oxm_class = OFPXMC_EXPERIMENTER */
        uint32_t experimenter;  /* Experimenter ID which takes the same
                                 * form as in struct ofp_experimenter_header. */
        uint8_t type;           /* Spatial mode type. One of  MMF, FMF, MCF */
        uint16_t id;            /* Spatial mode identifier */
};
```

5.1.5 Composite port match

This match type describes an ordered list of ports carrying the spectral-spatial signal. The encoding has a variable payload length and follows the same mechanism as the composite spectral flow match

```
/* Composite port match field.
 * The OXM_HEADER is followed by four bytes indicating the number of ports. */

#define OXM_EXP_CPORT OXM_HEADER (OFPXMC_EXPERIMENTER,OFPXMT_EXP_CPORT,4)

/* OFPXMT_EXP_CPORT_SIGID Payload format */
struct ofp_oxm_exp_CPORT_sigid {
        uint32_t oxm_header;    /* oxm_class = OFPXMC_EXPERIMENTER */
        uint32_t experimenter;  /* Experimenter ID which takes the same
                                 * form as in struct ofp_experimenter_header. */
        uint32_t port;          /* Port identifier. Refers to the OCh connection point */
};
```

5.1.6 Output action

Any change of the wavelength or frequency slot attributes of an optical signal requires a termination function implementing 3R signal regeneration. Therefore, switching a composite spectral-spatial signal requires the action directive to identify out-port and composite out-spatial mode set only. The specific encoding reuses the composite spatial match criteria defined in the previous sections. It may refer to a single OCh signal, port or spatial mode by reusing any number of *OFPXMT_EXP_OCH_SIGID, OFPXMT_EXP_CPORT_SIGID or OFPXMT_EXP_SM_SIGID* match fields.

5.2 Optical interface class instantiation

At the elementary spectral-spatial flow termination point, signal characteristics such as modulation format or forward error correction cannot be derived from information carried in a FLOW_MOD message. In order to program signal characteristic data at the connection endpoint the document proposes to extend the PORT_MOD message which is already designed to carry a variable list of port properties. OTPE already defines an optical transport property providing the container for a variable list of optical port features and a single OCh type port feature encoding the Optical Interface Class defined by its application code:

```
/* Optical Interface Class Feature Encoding */
struct ofp_port_optical_transport_application_code {
        uint16_t feature_type;      /* Set to OFPOTPF_OPT_INTERFACE_CLASS */
        uint16_t length;            /* length of feature excluding padding*/
        uint8_t oic_type;           /* OFPOICT_* identifies the relevant standard reference
                                     * specifying the application code */
        char app_code[15];          /* Null-terminated - Valid format/content for this field
                                     * is as per the relevant ITU-T standard indicated by
                                     * oic_type field or proprietary */
};
```

In order to support private application codes, the data definition should be changed to include the enterprise scope. A port may support multiple application codes and therefore may be characterized by multiple instances of the interface class feature. The 'active' flag indicates the currently active application code. This flag informs the SDN controller about the currently active application code, or indicates the desired application code as part of a connection creation.

```
/* Optical Interface Class Feature Encoding */
struct ofp_port_optical_transport_private_interface_class {
        uint16_t feature_type;      /* Set to OFPOTPF_OPT_PRIVATE_INTERFACE_CLASS */
        uint16_t length;            /* length of feature excluding padding*/
        uint32_t enterprise;        /* Private enterprise number */
        char app_code[15];          /* Private application code */
        uint8_t active;             /* TRUE, FALSE */,
};
```

6 Data Model

Up to this point we have focused our discussion on extensions to the OpenFlow-switch protocol that enable it to control an OpenFlow switch implemented on hardware that supports spectral-spatial switching as described in Section 3. In terms of the basic architectural components of SDN as shown in Figure 4-1 the OpenFlow switch protocol is (one) example of a D-CPI protocol.

The D-CPI is the lowest level of the SDN protocol stack and we have documented our proposed extensions in the same way ONF defines the protocol itself, i.e as 'C' language statements. For the purpose of our discussion we can consider that the protocol objects are 'modelled' as C structures. Introducing 'modelling' in this way allows us to recognize that the SDN community makes significant use of YANG and UML as data and information modelling tools respectively. The distinction between data and information modelling is subject to some debate, see for example [10], but is not especially important here. For our purposes it is sufficient to understand where those technologies are applied and by whom.

The ONF uses UML to define its Common Information Model (ONF-CIM) [11]. At the centre of ONF-CIM, the CoreModel fragment describes a technology independent information model partitioned into several packages. So far, two technology independent packages have been published by ONF: A CoreNetworkModule and a CoreFoundationModule. The two packages describe basic network aspects such as termination, forwarding and topology functions and common items such as identifiers and naming. Technology specific packages are expected to refine the data model. We anticipate developing a technology specific package to model spectral-spatial switching later in the Safari project.

In contrast to the broad scope of the ONF's UML based CIM, YANG is typically used to model data for a more restricted application or interface. In the context of this work YANG's particular significance is that it is employed as the modeling technology for a large number of commercial and community driven SDN controller implementations. Many of them implement OpenFlow-switch protocol embedded in a framework which supports various types of interfaces for extension and integration purposes following SDN architecture requirements. At its southbound interface, they typically implement common network access, monitoring and configuration protocols such as OpenFlow, NETCONF, HTTP, BGP, PCEP, SNMP, TL1 and others. The controller core itself provides an extensible execution environment for framework infrastructure, security and network service functions. A service abstraction layer offers common functions such as messaging and data storage and provides abstraction towards the southbound interface. The northbound interface usually offers integration options for business applications based on internet technologies such as REST [12]. Controller data models often make use of YANG as the modelling language. E.g. the community project Open Daylight [13] implements a model driven architecture targeted to simplify introduction of new functions starting with their YANG data model description. The same modelling language is also often used to describe controller external interfaces.

The Safari demonstrator will make use of data model and interface options deployed by community based SDN controllers to control programmable hardware. In general the flow extensions proposed in this document are applicable to any SDN controller implementation such as Open Daylight [13], ONOS [14] or Floodlight [15]. However, Open Daylight presents a SDN controller platform which is supported by large portion of the networking industry. Furthermore, a number of commercial SDN controllers managing optical transport devices, provide integration options into Open Daylight.

In the previous section, the document proposed extensions to OpenFlow-switch protocol in support of spectral-spatial flows. However, OpenFlow-switch was designed to deal with high packet rates – a feature of lesser importance to optical applications for which performance often is limited by end-to-end power levelling. Instead of implementing OpenFlow-switch as the protocol transporting control information, the demonstrator will deploy NETCONF to configure spectral-spatial flows. NETCONF [9] is an IETF network device configuration mechanism offering a remote procedure call context to network management applications. It uses Extensible Mark-up Language (XML) for configuration information representation. An alternative configuration protocol for the demonstrator could be RESTCONF which inherits the data store concept from NETCONF but uses HTTP as transport protocol and YANG as data definition language. As such, RESTCONF is well suited for integration into Web applications.

Open Daylight implements a node inventory model augmented by modules adding features such as flow control and management. The FlowNode inventory model augments flow objects representing ports, flow tables, instructions and counters for statistic metering. The flow data model is closely related to OpenFlow-switch protocol definition. These modules provide the pattern and starting point which will guide the demonstrator data model definition. In a first step the flow data model will be extended to include attributes and constructs needed to represent OCh flows. Safari demonstrator specific extensions such as multi-core identification, cross-talk monitoring objects and programmable modulation formats will enable the data model finally to operate spectral-spatial flows.

7 Conclusions

In order to operate optical connections across programmable multi-core capable hardware in the Safari testbed, we investigated abstractions to the OpenFlow control concept. We proposed a set of extensions permitting creation and deletion of wavelength connection across multi-core spatial components. Broadening the scope, we discussed information and data models suited to guide the demonstrator implementation and suggest a pragmatic approach which reuses OpenFlow compatible data model definitions deployed by community based SDN controllers such as Open Daylight.

REFERENCES

[1] Optical Transport Protocol Extensions 1.0, ONF, March 2015, https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/Optical_Transport_Protocol_Extensions_V1.0.pdf

[2] Concept paper of INSPACE project, Deliverable 7.1, INSPACE, February 2015, http://www.ict-inspace.eu/images/deliverables/4fc17e65-c0a5-4c68-b423-d1bb2601dfb4.pdf

[3] T. Mizuno, H. Takara, M. Oguma, T. Kobayashi, and Y. Miyamoto, "Modal Crosstalk Measurement Based on Intensity Tone for Few-Mode Fiber Transmission Systems," in Proc. OFC2014, W3D.5, 2014.

[4] SDN architecture 1.0, ONF, June 2014, https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf

[5] OpenFlow Switch Specification 1.4.0, ONF, October 2013, https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf

[6] OpenFlow Management and Configuration Protocol 1.2, ONF, 2014, https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1.2.pdf

[7] Network Configuration Protocol (NETCONF), Request for Comments 6241, IETF, June 2011, https://datatracker.ietf.org/doc/rfc6241/

[8] Charter: Optical Transport Working Group Charter, ONF, April 2013, https://www.opennetworking.org/images/stories/downloads/working-groups/charter-optical-transport.pdf

[9] Spatial-Spectral Flexible Optical Networking, INSPACE, FP7 ICT, October 2013, http://www.ict-inspace.eu/images/presentations/INSPACE_FactSheet.pdf

[10] Framework for Deriving Interface Data Schema from UML Information Models, Internet-Draft, March 2016, https://datatracker.ietf.org/doc/draft-betts-netmod-framework-data-schema-uml/

[11] Core Information Model (CoreModel) 1.1, ONF, November 2015, https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/ONF-CIM_Core_Model_base_document_1.1.pdf

[12] Architectural Styles and the Design of Network-based Software Architectures (Doctoral dissertation, Roy T. Fielding, 2000), http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[13] Open Daylight community Web-site, http://www.opendaylight.org/

[14] Open Network Operating System, http://onosproject.org/

[15] Floodlight, http://www.projectfloodlight.org/

List of Abbreviations

DSP	Digital Signal Processor
ONF	Open Networking Foundation
OIF	Open Internetworking Forum
SDN	Software Defined Network
PoC	Proof of Concept
MC	Media Channel
OTS	Optical Tributary Signal
CVNI	Control Virtual Network Interface
MIMO	Multiple-input multiple-output
ROADM	Reconfigurable Optical Add Drop Multiplexer

Document History

Version	Date	Authors	Comment
	15/02/2016	U. Häbel	Initial draft
	27/03/2016	T. Inui	Added “Use-case and required functionality”
	05/04/2016	T. Inui	Second draft
	06/04/2016	T. Morioka	Review of second draft