12-2008

An Object-Oriented Approach for Modeling and Simulation of Crack Growth in Cyclically Loaded Structures

D. Cojocaru
University of Delaware

Anette M. Karlsson
Cleveland State University, a.karlsson@csuohio.edu

Publisher's Statement

NOTICE: this is the author's version of a work that was accepted for publication in Advances in Engineering Software. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in Advances in Engineering Software, 39, 12, (12-01-2008); 10.1016/j.advengsoft.2008.01.009

Original Citation

Cojocaru, D., and Karlsson, A. M., 2008, "An Object-Oriented Approach for Modeling and Simulation of Crack Growth in Cyclically Loaded Structures," Advances in Engineering Software, 39(12) pp. 995-1009.

An object-oriented approach for modeling and simulation of crack growth in cyclically loaded structures

D. Cojocaru, A.M. Karlsson *

Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, United States

1. Introduction

Failures of structures subjected to cyclic loading are often driven by a slow evolution of structural properties, including changes in material properties and accumulation of damage. This is generically referred to as *fatigue of materials* [1]. The complexity and the interaction of the various factors influencing the lifetime of a cyclically loaded system make life prediction models difficult to build and define. To improve the design of a structure, the finite element method (FEM) is commonly employed to assess the contribution of various factors to the overall evolution of the structure [2,3]. Finite element analyses (FEA) typically give "snapshots" of the stress and strain fields at certain times, as well as information on how these fields change during one load cycle. However, the long computational time required to simulate a large number of cycles and to simulate the accumulation of damage, including cyclic crack growth, limits the usefulness of FEA for predicting fatigue behavior. An approach to reduce the computational time for the cyclically loaded structures is the so called *cycle-jump technique*, successfully applied in selected cases, e.g., [4–6]. In this paper, we will develop a technique for incorporating the accumulation of damage associated with fatigue, that is, *cyclic crack growth*. The most important attribute of the technique proposed in this paper is that crack propagation rate is *not* prescribed but predicted. The program decides-based on user-defined crack propagation criteria – when and how far the crack will propagate. That is, the crack will propagate once a critical value is reached, which may be a quantity that accumulates over multiple cycles.

The degradation associated with fatigue evolution can be divided into several stages [1], including nucleation of microcracks, growth and coalescence of microcracks in to larger cracks, and finally stable followed by unstable crack growth leading to final failure. Even though the flaw nucleation can be modeled through appropriate constitutive behaviors, e.g., utilizing theories within the field of Damage

* Corresponding author. Tel.: +1 302 831 6437; fax: +1 302 831 3619.
E-mail address: karlsson@udel.edu (A.M. Karlsson).

Mechanics [7], the simulation of crack propagation and coalescence using FEA still remains very challenging and computationally demanding.[1]

Several procedures have been proposed for simulating crack propagation in the context of FEM [8]. There are two main categories [8]: (i) the discontinuities in the model (i.e., cracks, voids) are represented geometrically (e.g., node release techniques), and (ii) the geometrical modeling of the defects is not required. In the latter case, the discontinuities are accounted for by using either an appropriate constitutive response [9] (e.g., "smeared crack"), or a modified displacement approximation [10] (e.g., extended finite element method [11]). An alternative approach combining aspects of the two categories above is represented by the cohesive zone model (CZM), which in recent years have become increasingly popular for predicting failure of material interfaces or crack propagation along know paths [12]. The behavior of the interface is described by a layer of cohesive elements via phenomenological laws called *traction-separation* or *cohesive* laws. Various traction-separation laws have been proposed in literature and a review is given in [13]. These laws include material dependent parameters requiring numerical or experimental calibration. One can distinguish between continuum cohesive zone model (CCZM) using cohesive FE elements based on a continuum formulation, and discrete cohesive model zone model (DCZM) which models the interface behavior via one-dimensional link or spring elements [14]. Cohesive elements can be problematic for cyclic loadings since the "failed" elements must have zero stiffness when the crack opens but must prohibit overlapping of the crack faces during crack closure. To overcome this challenge a contact formulation must be used either in addition to the cohesive elements (e.g., [15]) or included in the cohesive law [13]. Although, ABAQUS has built-in capabilities of modeling with cohesive elements [15], we do not use them in the context of the presented approach.

In this paper, we present a general modeling frame using a "node release" approach designed to simulate the evolution of the crack propagation for structures under cyclic loads, predicting the crack growth rate. The approach allows for arbitrary shaped flaws (e.g., cracks, voids), which makes our approach more general than when using cohesive elements. The modeling approach can be applied to single or multi-material systems. The crack propagation is mimicked by releasing nodes, previously tied together, when a *user-defined propagation criterion* is fulfilled. Our approach is unique in that the incremental cyclic crack extension is *not* defined by the user, instead other criteria – such as dissipated energy in the vicinity of the crack-tip – is used to determine *when* and *how far* a crack propagates. The technique requires the fracturing path to be known beforehand, capturing the behavior of a large number of cases, including multi-layered systems where the cracks preferentially propagate in the interface.

Our approach is to utilize the commercial software ABAQUS [15] and its object-oriented programming (OOP) interface. A considerable amount of work has been published in the last 20 years where OOP concepts are utilized for developing finite element codes: In a bibliographical review by Mackerle [16], almost 400 references on various OOP aspects of FEM are listed, ranging from OOP philosophy, OOP-FE code libraries to OOP-FE applications. Considerations and interesting examples of general FE codes implementation and development via OOP languages can be found for example in [17–23]. FE-based codes capable of simulating crack propagation exist (e.g., ADAPCRACK3D [24], FRANC3D [25], WARP3D [26]). However, to the knowledge of the authors, the literature is void on OOP-based procedures for crack growth in cyclically loaded structures [16].

The frame presented here leads to a fully parametric FE model that allows the user to implement crack propagation criteria based on any quantity available in the model (e.g., stress, strain, energy etc.). Another unique feature is that the propagation criteria are assigned to the end vertices of the intact segments of the interfaces. In this way, the necessity of explicitly modeling an initial (i.e., pre-existing) crack in order to simulate the propagation is eliminated. We will illustrate the use of a propagation criterion by using dissipated energy in the vicinity of the crack tip (see Section 4). The dissipated energy can be directly related to the accumulation of plastic strain, which in turn has been related to fatigue crack growth in metals [1].

The outline of the remainder of this paper is as follows: We first discuss the main concepts underlying the developed FE-based framework that can simulate cyclic crack growth (Section 2), and in Section 3 we will address implementation aspects. In Section 4, we discuss a particular propagation criterion used to illustrate the concepts (i.e., dissipated energy in the vicinity of the crack-tip), elucidating the versatility of the developed framework. This is applied in two selected benchmark problems discussed in the end of Section 4.

2. Modeling concepts

We selected to utilize the commercially available software ABAQUS [15] in our approach, with the hope of making the developed routine easier for other users to adopt. The developed code, which consists of a set of classes, is written in Python language and uses ABAQUS Scripting Interface (ASI) [27,28]. ASI is an object-oriented extension library based on Python [29], for advanced pre- and post-processing tasks of ABAQUS.

The procedure developed assumes an alternative model description to what is usually used in ABAQUS/CAE. Based on this approach, the ABAQUS model is generated automatically (Fig. 1A). The frame utilizes the concept that

[1] We note that when simulating cyclic stress and assuming linear-elastic, perfectly-plastic materials, the stress does not increase with each cycle, but will cycle between a maximum and minimum value. Thus, stress alone is not suitable as a criterion for fatigue crack propagation.

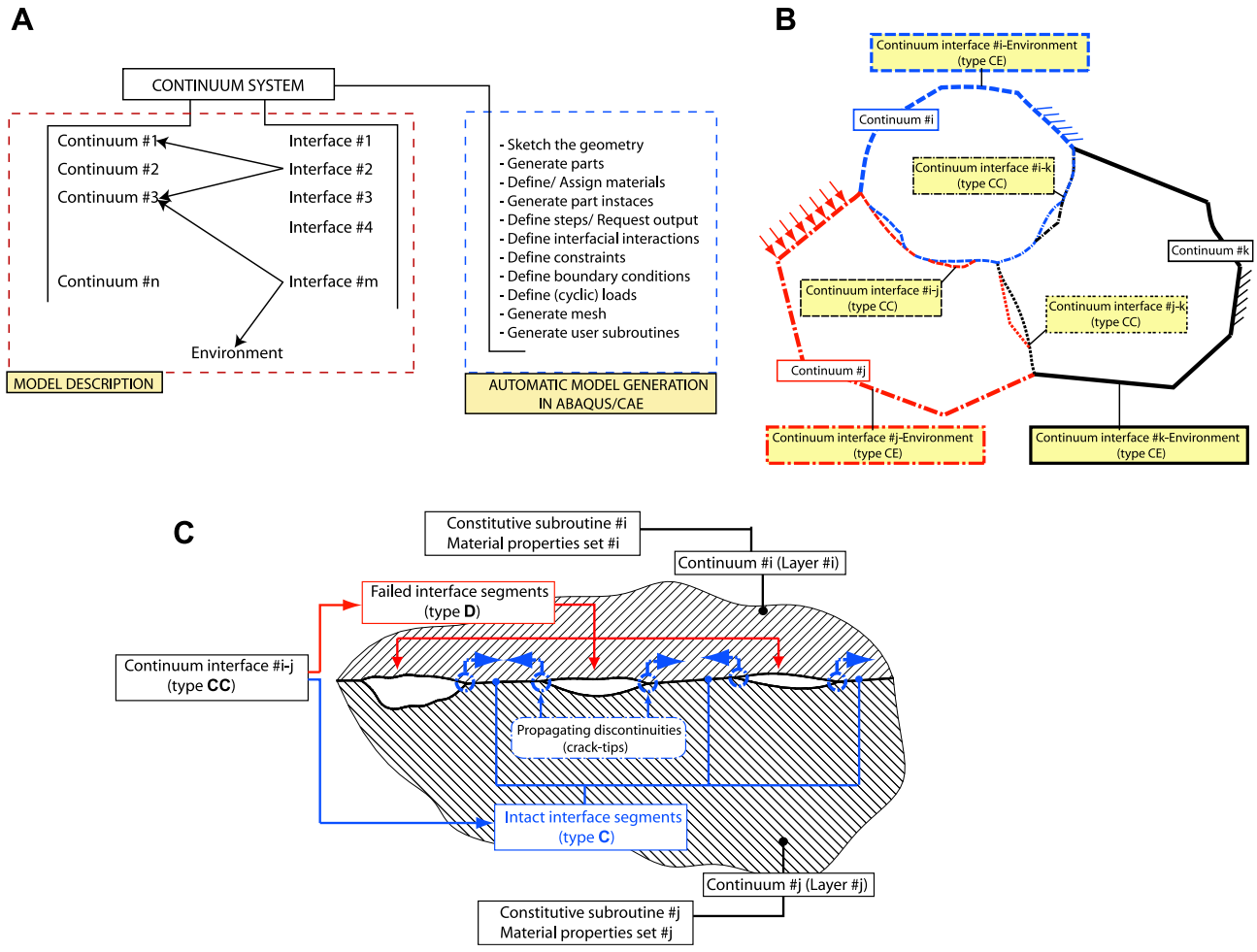

Fig. 1. Model description: (A) continuum system architecture; and (B) continuum system decomposition; (C) interface decomposition into failed and intact segments.

any two-dimensional (2D) multi-material system can be described by decomposing it into two sets of components: (i) *a set of continua* and (ii) a *set of continuum interfaces*. In the following, the overall assembly of continua and interfaces is referred to as a *continuum system* (Fig. 1A). This decomposition of information aims to take advantage of the object-oriented programming featured by ABAQUS Scripting Interface (ASI) [27,28]. Details on OOP can be found in many programming textbooks (e.g., [30]).

In our approach, a *continuum* represents a 2D subdomain of the complete model. A continuum has its own material description[2] and meshing related parameters (e.g., element type, meshing algorithm, section thickness). A *continuum interface* describes either: (i) how two continua interact with each other, i.e., interface of type continuum–continuum, CC (Fig. 1B); or (ii) how a continuum interacts with the exterior, i.e., interface of type continuum–environment, CE. Furthermore, a continuum interface is described by a sequence of *interface segments*, each segment characterizing a portion of the interface. For the CC interface (Fig. 1C), two types of segments are necessary, where segments of type D are used for modeling interface discontinuities and segments of type C for modeling continuous portions (e.g., crack ligaments). The failed interface segments can be used to model geometrically not only cracks but also flaws of various shapes at the interfaces (e.g., voids). This cannot be accomplished with cohesive elements. The sequence of discontinuous segments (type D) and intact segments (type C) in a CC interface can be arbitrary which makes the approach very useful for modeling arrays of flaws. A third type of interface segments, type S, specific to the CE continuum interfaces is intended to model the interface between a continuum and the exterior. The main characteristics of the interfaces and of the interface segments are summarized in Tables 1 and 2, respectively.

The 2D geometry of the continuum interfaces is assumed to be piece-wise linear. Thus, the type CC interfaces can be described in a discrete manner by two sets of

[2] A material description implies a set of properties and a routine responsible for the constitutive response. Here, the constitutive response associated with each continuum is implemented in the UMAT user subroutine.

Table 1
Interface types

Interface type	Modeling purpose	Geometric description
CC (continuum–continuum)	• Interior boundaries • Interactions between two neighboring continua	Two sets of points: one set for each continuum boundary
CE (continuum–environment)	• Exterior boundaries. • Interactions between a continuum and the exterior	One set of points

Table 2
Interface segment types

Segment type	Interface type	Modeling purpose	Features
D	CC	Interface discontinuities such as gaps, cracks, voids	• It can generate contact modeling requests for the separated edges of the interface it describes
C	CC	Intact portions of the interface	• It generates a *tie* constraint between the two sides of the interface • Other constraints can be implemented • The end points of the segment can be considered potential crack-tips. These points may have associated propagation criteria
S	CE	Portions of exterior boundaries	• It can have associated boundary conditions and cyclic loadings

points, whereas the geometric description of interfaces of type CE needs only one set of points (Table 1).[3] The geometric description is stored at the interface segment level, each interface segment being responsible for describing a specific portion of the interface. Thus, each interface segment contains a subset of points, describing a separate portion of the interface. At each of the points describing the interfaces, a node will be automatically created. The potential crack-tips in the model correspond to the end vertices of the *intact* interface segments (i.e., segments of type C). With this approach we eliminated the necessity of modeling a pre-existing crack and created a base for crack nucleation. Sets of user-defined propagation criteria can be assigned to the potential crack-tips. These criteria are evaluated iteratively after user prescribed intervals of cycles, in order to obtain the propagation increment for each crack-tip.

The possibility of implementing custom and time-dependent propagation criteria is very useful, especially in the context of simulating the failure of multi-layer structures, where the interfacial decohesion may be influenced by multiple factors [31,32]. The key feature is that the developed modeling frame allows for crack propagation due to cyclic loading.

[3] This geometric description assumes that any 2D curve can be represented accurately by supplying a sufficient number of points.

3. Implementation

The modeling frame has been developed in Python and makes use of ABAQUS Scripting Interface (ASI). The implementation is decomposed into three main levels (Fig. 2): (i) Python level, (ii) ABAQUS level and (iii) FORTRAN level. The Python code controls the entire computational process. At the ABAQUS level, we included the pre-/post-processor ABAQUS/CAE and the ABAQUS Solver (ABAQUS/Standard). The ABAQUS/CAE pre-/post-processor contains an integrated Python interpreter, which executes the code mentioned at the Python level (Fig. 2). The ABAQUS Solver utilizes the ABAQUS user subroutines (FORTRAN level) when needed, for example to impose a user specified displacements field or for the constitutive response. These necessary user subroutines are generated automatically by the Python code, based on predefined subroutine templates stored as text files.

By implementing the modeling concepts in a separate programming "layer" the proposed modeling frame ensures the generality for future developments. As an example, a new type of continuum interface can be implemented in the framework to model the continuum interfaces via cohesive elements as an alternative to the approach described in this paper.

The model description in terms of continua and interfaces (i.e., using the associated classes discussed below) is coded by the user at the Python level (Fig. 2). Cyclic loads can be defined and assigned to both continua and interface segments. Automatic meshing can be used for portions of the continua, based on the existing ABAQUS meshing algorithms. In addition, the user can prescribe the number of cycles to be solved before the interfaces will be updated (updated based on the propagation criteria). By programming the model description, the modeling is done in a fully parametric manner. Once the model is described, it can be entirely generated in ABAQUS/CAE together with the associated user subroutines by the Python code (see the class CContinuumSystem, below).

A simple class (CAnalysis) has been developed to control the overall computational process. This class employs the automatic generation of the ABAQUS model at the beginning of the computational process, and then submits the model to be solved by the ABAQUS Solver. Once the FE solution is obtained, the CAnalysis object iteratively evaluates the propagation criteria and updates (if needed) the continuum interfaces. During each iteration, the geometry (i.e., the sequence of points defining the geometry) of the interface segments is modified if any assigned criterion is fulfilled. After the iterative updating, the current description of the continuum interfaces is saved to an external file. Further, the CAnalysis object outputs the solution to be

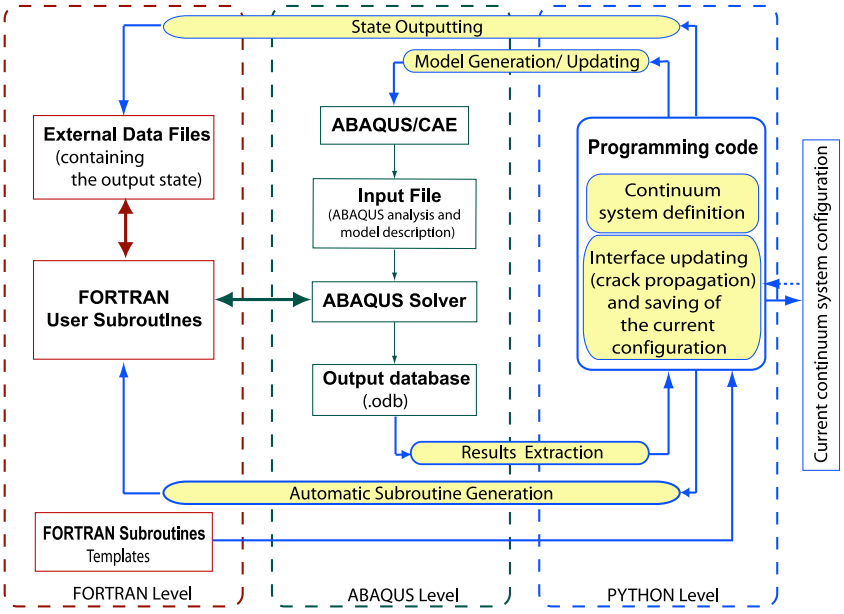

Fig. 2. Schematics of implementation in the context of ABAQUS.

used as the initial state in the next analysis and re-submit the model for further solving. This procedure is repeated until a total number of cycles prescribed by the user is reached. In this manner, the solver and the pre-/post-processor program (i.e., ABAQUS/CAE) only deal with small output database files, eliminating the increased solution time associated with the management of large output files.

The classes are grouped in two distinct modules, Fig. 3A: (i) module cAnalysis.py which contains only the class CAnalysis and (ii) continuumClasses.py which gather the definitions of all classes contributing to model description. The main classes are described in the following and their most important attributes and methods are shown in Fig. 3B.

Class CAnalysis (defined in module CAnalysis.py) controls the overall computational process. It contains, as an attribute, a CContinuumSystem object which handles the modeling part of the analysis (see Fig. 3B).

Class CContinuumSystem (defined in module continuumClasses.py) stores a set of CContinuum and CContinuumInterface objects from where it generates the ABAQUS FE model or updates a previously generated model, using the ABAQUS Scripting Interface.

Class CContinuum (defined in module continuumClasses.py) is used to describe a continuum, i.e., a sub-domain of the structure characterized by a specific constitutive response. Its boundaries are described by a set of continuum interfaces (defined as CContinuumInterface objects), which separate it from the environment (interfaces of type CE) or from an adjacent continuum (interfaces of type CC).

Class CContinuumInterface (defined in module continuumClasses.py) implements the code necessary to describe an interface between two continua or a continuum and the environment. For simplicity, the two types of continuum interfaces (i.e., type CC and type CE) are implemented within the same class, i.e., CContinuumInterface. The type of the interface is stored in an attribute, *CContinuumInterface.type*, which is checked by any function which needs to distinguish between the two interface types.

Class CContinuumInterfaceSegment (defined in module continuumClasses.py) contains the code necessary for describing a segment of a continuum interface. The CContinuumInterfaceSegment objects are stored in the *segmentsList* attribute of a parent CContinuumInterface object. The interface segment of type C may have two associated CCrackTipOptions objects (one to each of its ends), which contain information about the propagation criteria.

4. Simulating cyclic crack propagation

4.1. Propagation criteria for cyclically loaded structures

4.1.1. General approach

We now discuss our approach for allowing a crack to grow due to cyclic loading. Our approach is unique in that the incremental cyclic crack extension is *not* needed to be defined (e.g., the commonly used crack extension per load cycle, da/dN is not needed); instead other criteria (such as dissipated energy in the vicinity of the crack-tip) can be used to predict the crack propagation rate. In fact, our modeling frame let the user implement (with little programming effort) custom crack propagation criteria, which can be based on any quantity available in the result database. In addition, each crack-tip can be assigned a different set of propagation criteria.

Commercially available FE-codes do not allow the users to build their own propagation criteria; when available the crack propagation is triggered by a classical fracture mechanics parameter. ABAQUS feature two built-in

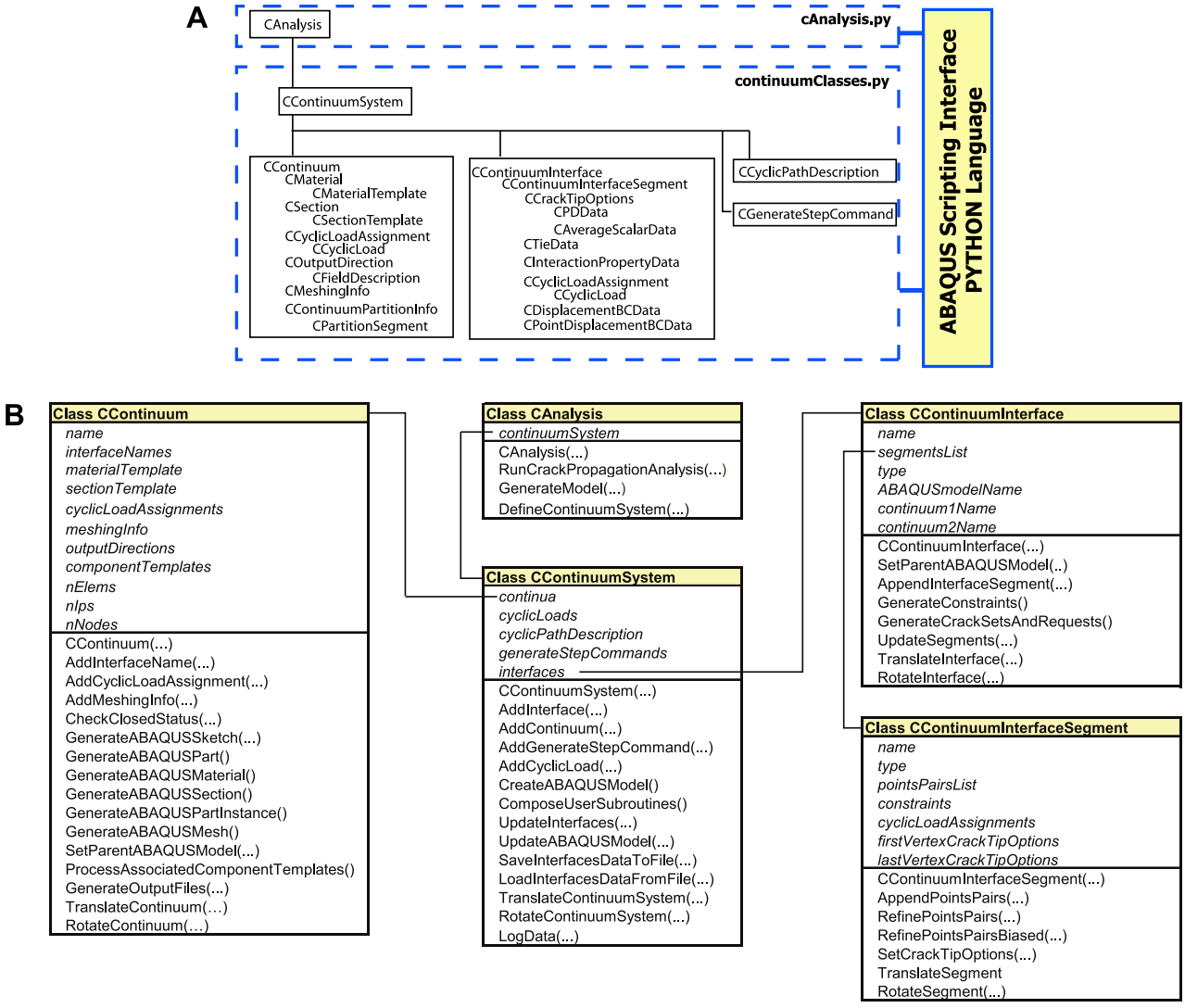

Fig. 3. Framework of implementation: (A) classes organization and (B) methods and attributes of the main classes.

propagation criteria: (i) a *critical stress criterion* – based on a stress-derived quantity at a point ahead the crack-tip, and (ii) a *crack opening displacement criterion* – established as the distance between two points behind the crack tip [15]. In general, none of these criteria are suitable when investigating fatigue, since they do not capture the *accumulation of damage* that eventually leads to fatigue crack growth. In our modeling frame, we do not use the built-in ABAQUS crack propagation criteria, but these could easily be implemented.

The propagation criteria presented here are assigned to the crack-tips through the CCrackTipOptions objects (Fig. 3A). In principle, a propagation criterion should consist of two parts: the first part (commonly expressed as an inequality) triggers the propagation, and the second part supplies the length of the crack increment, Δa. Two types of propagation criteria can be distinguished: (i) point-wise criteria (e.g., stress, strain at a point, crack-tip opening displacement), and (ii) integrated criteria (e.g., *J*-integral, strain energy, dissipated energy). In the current approach, the propagation increment, $\Delta a^i|_N$ after the *N*th loading cycle for any crack-tip *i,* is obtained by an iterative evaluation of the assigned propagation criteria.

The crack propagation criteria are evaluated in the end of a user-specified set of loading cycles.[4] A CCrackTipOptions object contains a heterogeneous collection of objects of different classes. Each of these objects implements a user-defined propagation criterion. Several classes have been developed so far (each one implementing a specific criterion), but the user can easily code additional classes implementing other propagation criteria if needed. A class implementing a propagation criterion should contain a method called IsCriterionFulfilled(...) This method should be given access to the FE results database and to the set of points describing the geometry of CContinuumInterface-

[4] Although the user is allowed to specify the time interval (as number of cycles) when the interface updating should be called, in the examples included here, the interface updating was requested after each FE computed cycle.

Segment objects and should return a Boolean value (i.e., TRUE or FALSE) as the criterion is deemed fulfilled or not. Examples of developed classes implementing propagation criteria are CPDData and CAverageScalarData (Fig. 3A). After a prescribed number of load cycles are simulated the CAnalysis.RunCrackPropagationAnalysis(...) method calls in turn the CAnalysis. *continuumSystem*.UpdateInterfaces(...) and CAnalysis.*continuumSystem*.UpdateABAQUSModel(...) methods (see Fig. 3B).

4.1.2. A criterion based on dissipated energy

To illustrate the implementation of propagation criteria, we introduce a new criterion based on the plastically dissipated energy integrated over a user specified domain in the vicinity of the crack-tip. The functionality of this criterion is encapsulated in the CPDData class (Fig. 3A). (Other energetic quantities could be used in a similar manner.) Cyclic yielding, which is mathematically linked to the plastically dissipated energy, is associated with the accumulation of dislocations, which is a major driving force in fatigue of metals. The plastically dissipated energy in the crack-tip vicinity and the possibility of using it as a crack driving parameter has been investigated by various authors [33,34]. Recent experimental approaches for characterizing the crack-tip plastic dissipation have been published in [35,36].

Due to the discrete nature of the FE model, the integration domain, D, (not to be confused with segment type D) is also discrete, and consists of a set of elements, $\mathbf{E}_D$, and nodes, $\mathbf{N}_D$, Fig. 4. The integration domain is positioned with respect to the crack-tip. Since the crack-tip can propagate along the continuum interface, so must do the integration domain. To accomplish this, we developed several functions for selecting the nodes and the elements in a region positioned with respect to the current crack-tip position.

Consider two continua A and B, separated by an interface (modeled through a CContinuumInterface of type CC). We construct a set of nodes, $\mathbf{N}_D$, according to the relation

$$\mathbf{N}_D = \mathbf{N}^A \cup \mathbf{N}^B \qquad (1)\leftarrow$$

where

$$\mathbf{N}^A = \{\text{Node} \in \Omega_A | f_A(Node) = \text{TRUE}\}\leftarrow \qquad (2a)\leftarrow$$

$$\mathbf{N}^B = \{\text{Node} \in \Omega_B | f_B(Node) = \text{TRUE}\}\leftarrow \qquad (2b)\leftarrow$$

Here, Ω_A and Ω_B represent the discretizations (meshes) of the two continua, and f_A and f_B represent the selection criteria for the nodes in the domain of continuum A and continuum B, respectively. For example, to obtain the nodes within a disk of radius R and center O, both selection criteria $f_{A,B}$ take the form $\|\mathbf{r}_{\text{Node}} - \mathbf{r}_O\| \leqslant R$, where $\mathbf{r}_{\text{Node}}$ and $\mathbf{r}_O$ denote the position vectors of the node and the disk center, respectively (Fig. 4A). Further, we obtain the set of elements:

$$\mathbf{E}_D = \mathbf{E}^A \cup \mathbf{E}^B \qquad (3)\leftarrow$$

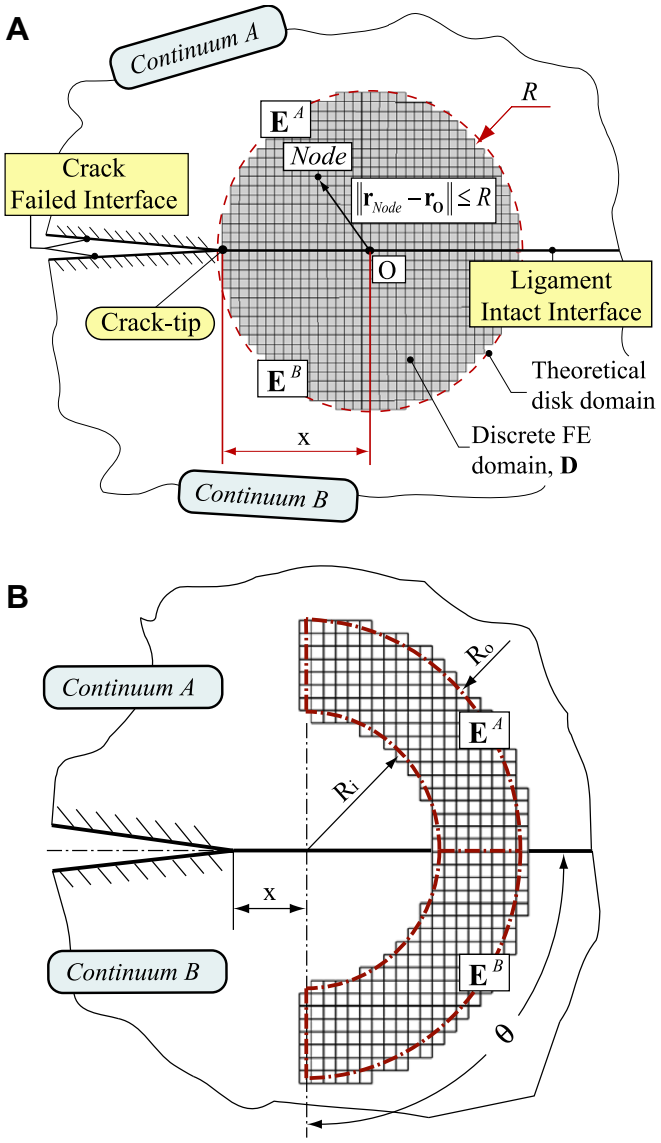

Fig. 4. Discrete integration domain for crack propagation criterion: (A) disk domain, $D(x;R)$; and (B) annular segment, $D(x;R_i,R_o,\theta)$.

where

$$\mathbf{E}^A = \{\text{Element} \in \Omega_A | n(\mathbf{N}^A \cap \mathbf{N}^{\text{Element}}) \geqslant n_A\}\leftarrow \qquad (4a)\leftarrow$$

$$\mathbf{E}^B = \{\text{Element} \in \Omega_B | n(\mathbf{N}^B \cap \mathbf{N}^{\text{Element}}) \geqslant n_B\}\leftarrow \qquad (4b)\leftarrow$$

where $\mathbf{N}^{\text{Element}}$ denotes the set of nodes belonging to a given element. The function $n(\mathbf{N}^A \cap \mathbf{N}^{\text{Element}})$ represents the number of nodes belonging to a given element which have been previously included in the node set $\mathbf{N}^A$ (Eq. 2a). The integer parameters n_A and n_B (in 4a and 4b) control the selection of elements and can take the values $1,2,\ldots,n(\mathbf{N}^{\text{Element}})$, where $n(\mathbf{N}^{\text{Element}})$ represents the number of the nodes of a element [e.g., for a bi-linear solid element $n(\mathbf{N}^{\text{Element}}) = 4$ whereas for a bi-quadratic serendipity solid element $n(\mathbf{N}^{\text{Element}}) = 8$].

The selection function returning the set of elements within the domain D (i.e., $\mathbf{E}_D$) is denoted $D(x;\alpha_i)$, where x characterizes the position of the domain D with respect

to the crack-tip along the interface and α_i represents a set of parameters describing the shape of the domain D. Two discrete domain selection functions were used for the applications presented in the benchmark problems: (i) $D(x;R)$ to obtain a discrete disk domain (Fig. 4A), and (ii) $D(x;R_i,R_o,\theta)$ to obtain a discrete annular segment (Fig. 4B). These two selection functions can also be used to select a rectangular region comprising 2 or 4 elements. For example, if h_e is the element size (in a region with 2D elements of constant size) then, if the function $D(x;R)$ is called with $R < h_e$, it will return a square domain comprising four elements.

From the discrete nature of the domain D, the integration simply becomes a summation over the elements included in the domain:

$$W^p(D) = \sum_{e \in \mathbf{E}_D} w_e^p \qquad (5)\leftarrow$$

where w_e^p represents the plastically dissipated energy over the domain of an element e. The value of w_e^p is computed by numerical integration of the density of dissipated energy, which in turn is supplied by the constitutive subroutine at each integration point.

The crack increment, $\Delta a|_N$, (after N cycles) is obtained by iterative updating of the continuum interfaces according to (considering one crack):

$$\bullet\leftarrow \Delta x = 0.0, k = 0 \qquad (6a)\leftarrow$$

$$\bullet\leftarrow \mathbf{E}_D \quad D(x + \Delta x; \alpha_i) \qquad (6b)\leftarrow$$

$$\bullet\leftarrow W^p(D) = \sum_{e \in \mathbf{E}_D} w_e^p \qquad (6c)$$

$$\bullet\leftarrow \text{while } W^p(D) \geqslant W^p_{\text{cr}} :- \qquad (6d)$$

$$\blacksquare \quad \Delta x = \Delta x + h_e, \quad k = k + 1 \qquad (6e)\leftarrow$$

$$\blacksquare \quad \mathbf{E}_D \quad D(x + \Delta x; \alpha_i) \qquad (6f)\leftarrow$$

$$\blacksquare \quad W^p(D) = \sum_{e \in \mathbf{E}_D} w_e^p \qquad (6g)\leftarrow$$

$$\bullet\leftarrow \Delta a|_N = \Delta x \qquad (6h)\leftarrow$$

In the above scheme, h_e represents the element size along the continuum interface, x is the fixed position of the domain with respect to the *current* crack-tip location (the domain moves with the crack-tip) and W^p_{cr} represents a threshold value of dissipated energy triggering the propagation. Although W^p_{cr} is assumed constant in the following examples, it can easily be made a function of other parameters such as the total crack length or the cycle number, i.e., $W^p_{\text{cr}} = W^p_{\text{cr}}(a, N)$.

Thus, the resulting crack extension will be discrete. The crack extension will depend on the shape and position of the integration domain, and on the mesh size. The calibration of such a criterion can be established only in the

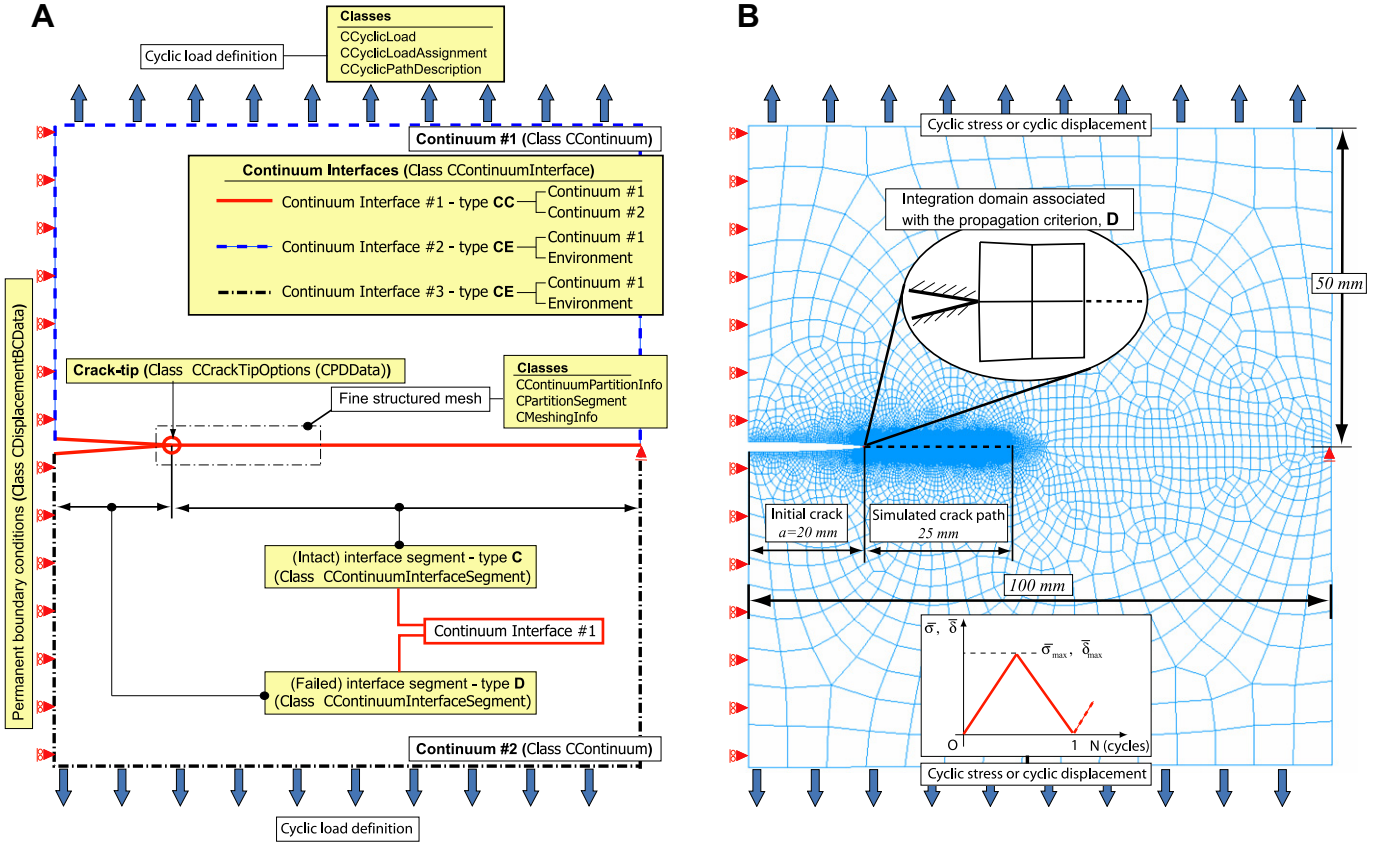

Fig. 5. Center crack model: (A) continuum system description; and (B) generated ABAQUS model.

context of experimental testing and will be address in a future study.

In the remainder of this section, two benchmark problems will be presented, to elucidate our proposed method.

4.2. Cyclic propagation of a center crack

4.2.1. Model description

First, we will analyze a plane-strain specimen with a centered crack, Fig. 5. Symmetry about the vertical axis (left side in Fig. 5) is assumed to reduce the model size. The initial half length of the crack is 20 mm, the half width is 100 mm and the total height is 100 mm. The model is described in terms of continua and continuum interfaces (Fig. 5A) based on which the ABAQUS model is generated automatically (Fig. 5B). Two load cases are considered: (i) cyclic stress and (ii) cyclic vertical displacement, applied to the top and the bottom edges of the specimen. Bi-quadratic plane-strain elements with reduced integration (CPE8R) are used for the entire domain. The generated FE model for this example contains 51,780 nodes and 16,966 bi-quadratic elements. The element size of the structured mesh near the crack path is $h_e = 0.1$ mm. The crack propagation is triggered by the plastically dissipated energy criterion (see Section 4.1). For simplicity, the integration domain

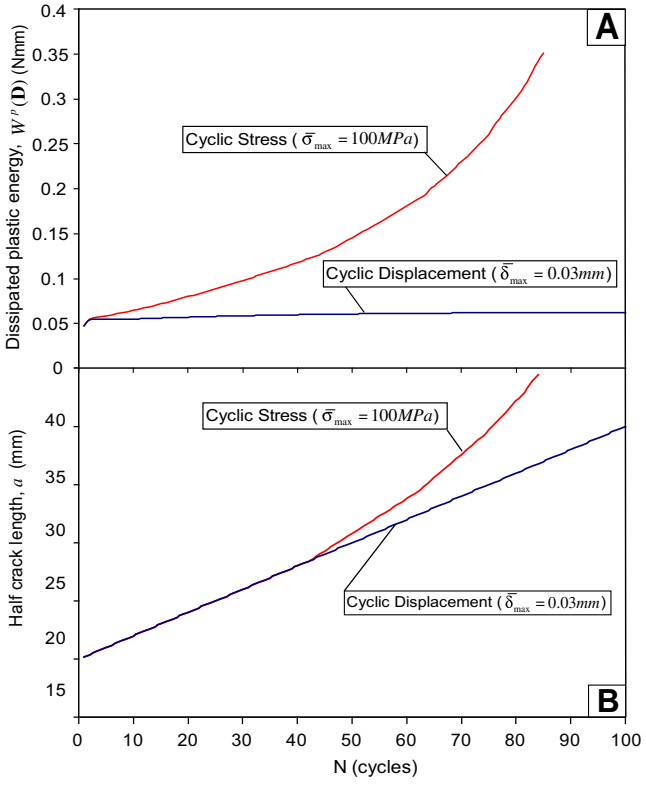

Fig. 6. For the center crack model, system evolution for applied cyclic stress ($\bar{\sigma}_{max} = 100$ MPa)←and applied cyclic displacement ($\bar{\delta}_{max} =$← 0.03 mm): (A) dissipated plastic energy, $W^p(D)$, in the domain ahead the crack-tip during the first iteration; and (B) evolution of the crack half length.

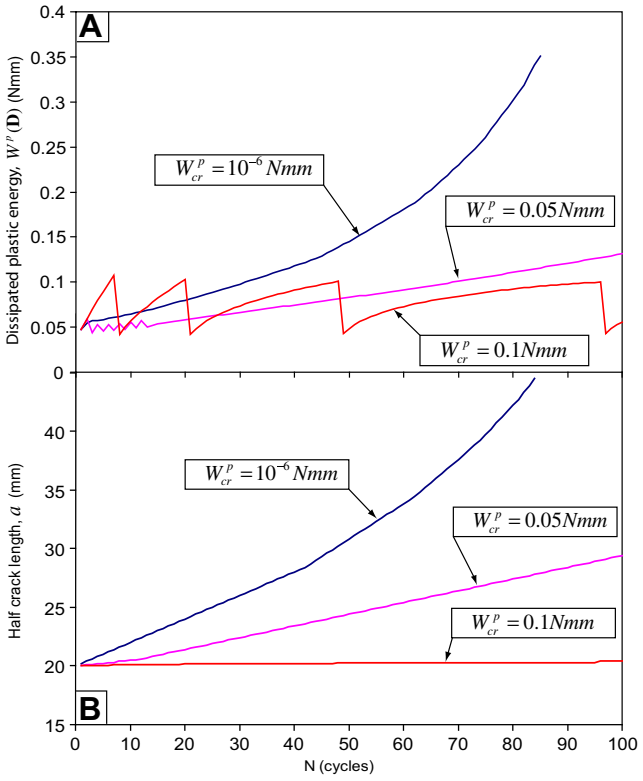

Fig. 7. For the center crack model, influence of the threshold value W_{cr}^p on the system evolution: (A) dissipated plastic energy, $W^p(D)$, in the domain ahead the crack-tip during the first iteration; and (B) evolution of the crack half length.

associated with this criterion is set to a square region in front of the crack (Fig. 5B). This region comprises four elements ahead the crack-tip and it is given by the domain selection function $D(x = h_e; R < h_e)$ (see Fig. 4A). To preserve the size and shape of the discrete integration domain, we used a structured mesh with constant element size in the vicinity of the crack, along the continuum interface separating the two continua. For the remaining of the specimen, we used a free meshing technique. The J_2 *computational plasticity theory* [37] was used to describe the constitutive response for both continua. For simplicity, the material is considered linear-elastic, perfectly plastic. The material properties are: elastic modulus $E = 210$ GPa, Poisson's ratio $v = 0.3$ and yield strength $\sigma_Y = 600$ MPa. The load cycle is decomposed into two steps: *loading*$(0 \rightarrow L)$ and *unloading*$(L \rightarrow 0)$, where L denotes the magnitude of the maximum cyclic load, either stress or displacement. The computational time for the following simulations has been estimated at approximately 18 min/computed loading cycle.[5]

4.2.2. Cyclic stress vs. cyclic displacement

First, we will investigate the crack propagation under two cyclic tensile conditions: (i) cyclic stress:

[5] Using a single core, 3.2 GHz CPU and 2 GB RAM memory.

$\bar{\sigma}_{\min} = 0$ MPa $\rightarrow\leftarrow \bar{\sigma}_{\max} = 100$ MPa $\rightarrow\leftarrow \bar{\sigma}_{\min} = 0$ MPa, and (ii) cyclic displacement: $\bar{\delta}_{\min} = 0$ mm $\rightarrow\leftarrow \bar{\delta}_{\max} = \leftarrow$ 0.03 mm $\rightarrow\leftarrow \bar{\delta}_{\min} = 0$ mm, of the top and bottom edges as indicated in Fig. 5B. In the second case, the maximum relative displacement between the top and bottom edges is 0.06 mm. The values of $\bar{\sigma}_{\max}$ and $\bar{\delta}_{\max}$ have been chosen such that the amounts of the dissipated energy in the integrated domain during the first loading cycle are equal. The threshold value of the propagation criterion, $W^p_{\text{cr}} = 10^{-6}$ N mm, was intentionally selected low, so that crack propagation occurs during each cycle. The simulation is conducted until the crack has propagated through the region of fine mesh (i.e., total half crack length is 45 mm, Fig. 5B).

The accumulated dissipated energy in the integrated domain, $W^p(D)$, is investigated after each cycle. Its evolution obtained during the first iteration (i.e., for $k = 1$ in Eq. (6e)) of the interface updating procedure (which runs in the end of each cycle) is shown in Fig. 6A. Recall that the integrated domain moves with the crack-tip as the crack propagates (see Eqs. (6a)–(6h)). For the case of cyclic displacement, the accumulated dissipated energy in the (moving) integrated domain, $W^p(D)$, is almost constant as the structure is cycled (Fig. 6A), resulting in that the crack propagates at a constant rate (Fig. 6B). Thus, for the displacement controlled cyclic loading the crack propagation rate appears independent of the crack length. For the case of cyclic stress, $W^p(D)$ increases monotonically and nonlinearly as the structure is cycled and the crack propagates. In other words, as the crack propagates (Fig. 6B), more energy dissipates on a cyclic basis in the domain D, ahead the crack-tip, which in turn will accelerate the crack propagation. After each cycle, the crack propagates at least one element length and $W^p(D)$ is higher in this new geometric location than for the previous location. This accelerated crack propagation agrees with typical results presented in the literature (e.g., [1]).

4.2.3. *Influence of the critical value for the case of cyclic stress*

We will next investigate the influence of the threshold value, W^p_{cr}, on the overall response. For the case of cyclic stress, three distinct values of the critical plastic dissipation, W^p_{cr}, were selected: 10^{-6} N mm, 0.05 N mm and 0.1 N mm, while all the other parameters were kept constant.

For the case $W^p_{\text{cr}} = 10^{-6}$ N mm (discussed above), the crack propagates for each cycle, and the crack will extend through the region of refined mesh after 84 cycles (i.e., $a > 45$ mm, Fig. 5B). The evolution for the case of $W^p_{\text{cr}} = 0.05$ N mm comprises two parts. In the first 13 cycles, the crack propagates intermittently since the propagation criterion is not fulfilled after each individual cycle. Once the criterion is fulfilled (i.e., $W^p(D) \geqslant W^p_{\text{cr}}$), propagation occurs and the integration domain D (which is posi-

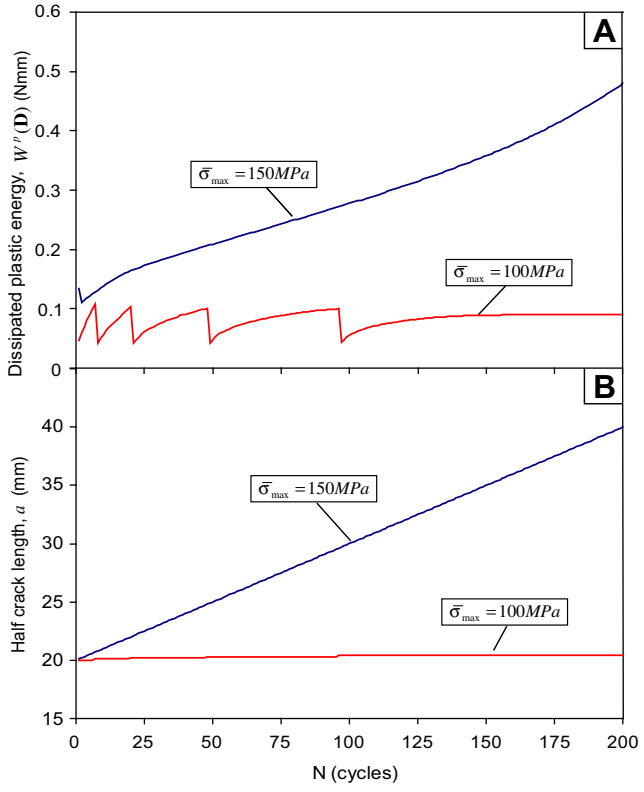

Fig. 8. For the center crack model, influence of the maximum applied stress on the system evolution: (A) dissipated plastic energy, $W^p(D)$, in the domain ahead the crack tip during the first iteration, and (B) evolution of the crack half length.

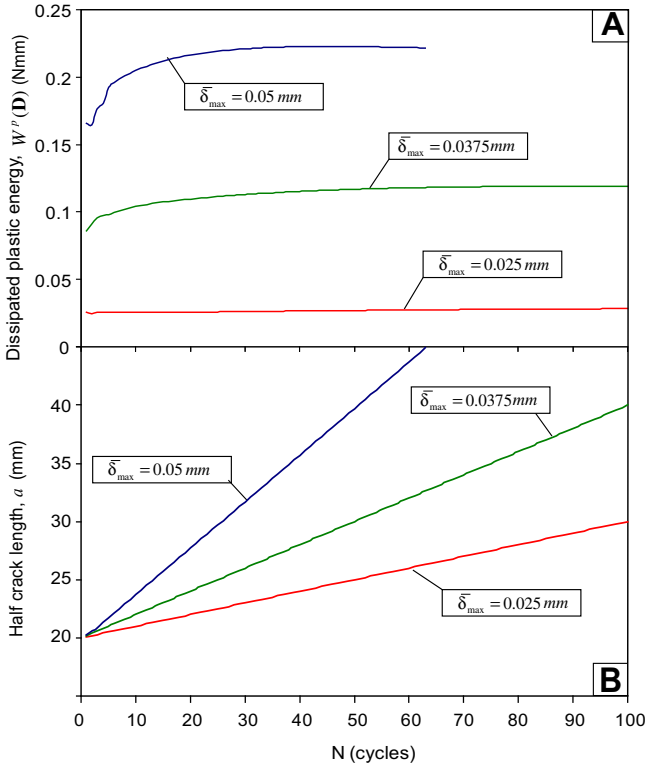

Fig. 9. For the center crack model, influence of the maximum applied displacement on the system evolution: (A) dissipated plastic energy, $W^p(D)$, in the domain ahead the crack tip during the first iteration, and (B) evolution of the crack half length.

tioned with respect to the current crack-tip) will encompass a region with a lower level of plastic dissipation (Fig. 7A). After the 13th cycle, the level of the dissipated energy is sufficient to trigger crack propagation after each of the computed cycles (Fig. 7B).

The case of $W^p_{cr} = 0.1$ N mm generates an interesting outcome. The propagation criterion requires the plastic dissipation to accumulate over a larger and larger number of cycles until the crack propagation criterion is fulfilled (Fig. 7A). The crack-tip propagates only four times during the 100 cycles investigated here (Fig. 7B).

4.2.4. *Influence of the maximum cyclic stress*

Next, we investigate the influence of the applied maximum cyclic stress on the overall behavior. Two simulations were performed: (i) $\bar{\sigma}_{max} = 100$ MPa and (ii) $\bar{\sigma}_{max} = 150$ MPa. All other model parameters were kept constant. The simulations span 200 cycles and the critical dissipated energy was set to $W^p_{cr} = 0.1$ N mm.

The case $\bar{\sigma}_{max} = 100$ MPa was discussed above (Fig. 7). When simulating 200 cycles (instead of the previous 100), we found that the crack arrests after 97 cycles (Fig. 8B) due to insufficient plastic dissipation in the integrated domain ahead the crack-tip (Fig. 8A). A literature review suggests that plasticity in the crack-tip vicinity can play a decisive role on crack growth through crack-closure mechanisms [38]. A detailed assessment is beyond the scope of the current article and will be addressed in a subsequent study.

When $\bar{\sigma}_{max} = 150$ MPa, the dissipated energy, $W^p(D)$, shows a monotonically increasing and slightly non-linear behavior (Fig. 8). For this case, the crack will propagate with the same crack extension $\Delta a|_N$ for all of the 200 computed cycles.

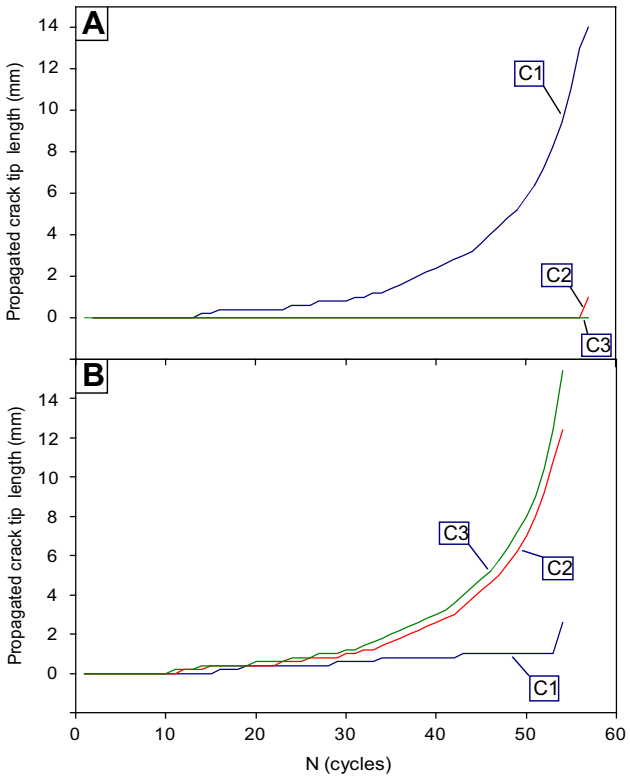

Fig. 11. For the bi-layered model, cyclic crack evolution: (A) for the case $a_2 = 15$ mm and (B) for the case $a_2 = 20$ mm.

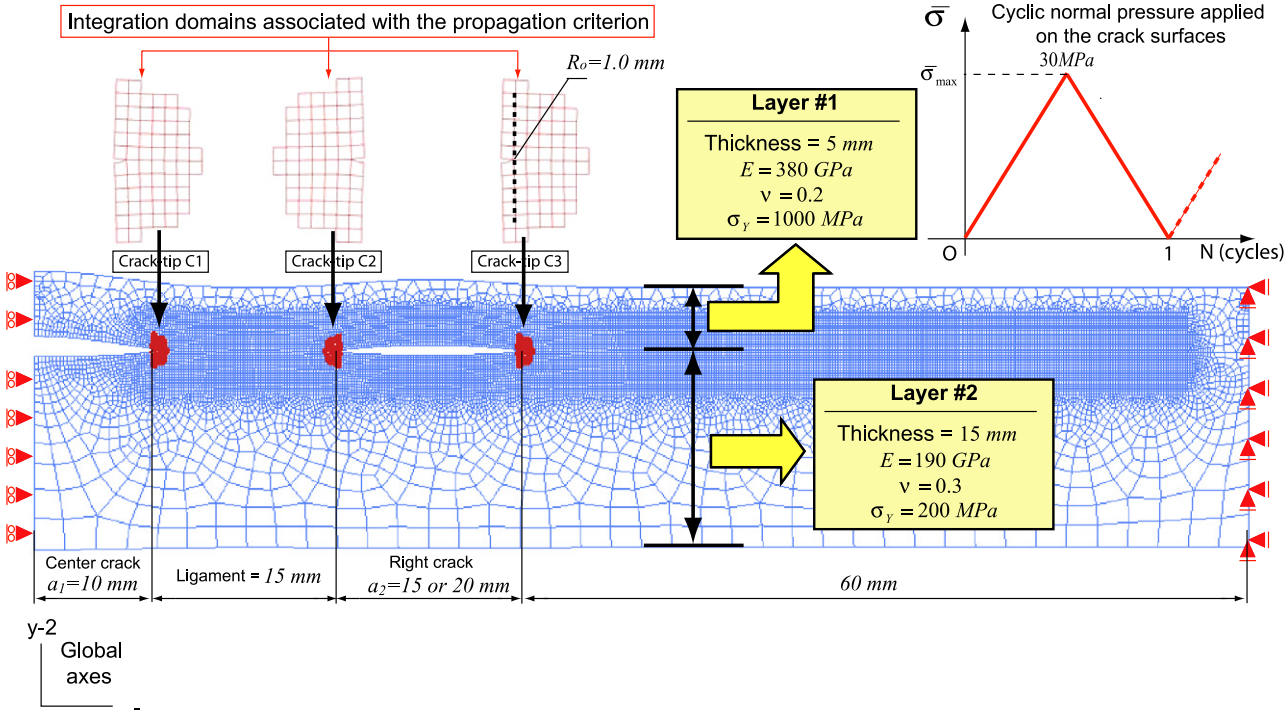

Fig. 10. Bi-layered model: the coating corresponds to layer 1 and the substrate to layer 2.

4.2.5. Influence of the maximum cyclic displacement

Lastly, we discuss the influence of the maximum applied cyclic displacement on the crack propagation. Three cases were simulated with (i) $\bar{\delta}_{\max} = 0.025$ mm, (ii) $\bar{\delta}_{\max} = 0.0375$ mm and (iii) $\bar{\delta}_{\max} = 0.05$ mm, where $\bar{\delta}_{\min} = 0$ mm $\rightarrow \bar{\delta}_{\max} \rightarrow \bar{\delta}_{\min} = 0$ mm. The simulations were conducted for 100 cycles or until the total half crack length reaches 45 mm. For all cases, the crack propagates with constant increments $\Delta a|_N$ (i.e., the evolution of the crack length is linear, Fig. 9B) independently of maximum applied displacement. Moreover, the dissipated energy in the integrated domain, $W^p(D)$, increases in the beginning of analysis and reaches a plateau. For $\bar{\delta}_{\max} = 0.05$ mm, the dissipated energy decreases slightly during the last five cycles which corresponds of a drop of less than 0.5%. This may be attributed to: (i) the structure becoming more compliant due to crack growth and/or (ii) the crack approaching the coarse mesh region (i.e., the half crack length approaches 45 mm).

4.3. Cyclic crack propagation in a bi-layer

We will next simulate the spallation of a coating from a substrate. The constitutive response is assumed linear-elastic, perfectly plastic for both layers, with properties given in Fig. 10. According to the modeling frame we developed, each layer is described as a continuum. The interface between the two layers includes *two cracks* separated by a 15 mm ligament (Fig. 10). The model assumes symmetry about the vertical axis at $x = 0$ (left side in Fig. 10). The center crack has the half length $a_1 = 10$ mm whereas the right crack is characterized by the length a_2 of either 15 mm or 20 mm. During each loading cycle, the *current interface* surfaces of the two cracks are subjected to applied cyclic pressure: $\bar{\sigma}_{\min} = 0$ MPa $\rightarrow \bar{\sigma}_{\max} = 30$ MPa $\rightarrow \bar{\sigma}_{\min} =\leftarrow$ 0 MPa. The size of the elements along the layers interface is $h_e = 0.2$ mm (i.e., the discrete increment of interface decohesion will be $\Delta a|_N = 0.2k$, $k = 0, 1, 2, \ldots$). The generated FE model has 50,399 nodes and 17,771 bi-quadratic

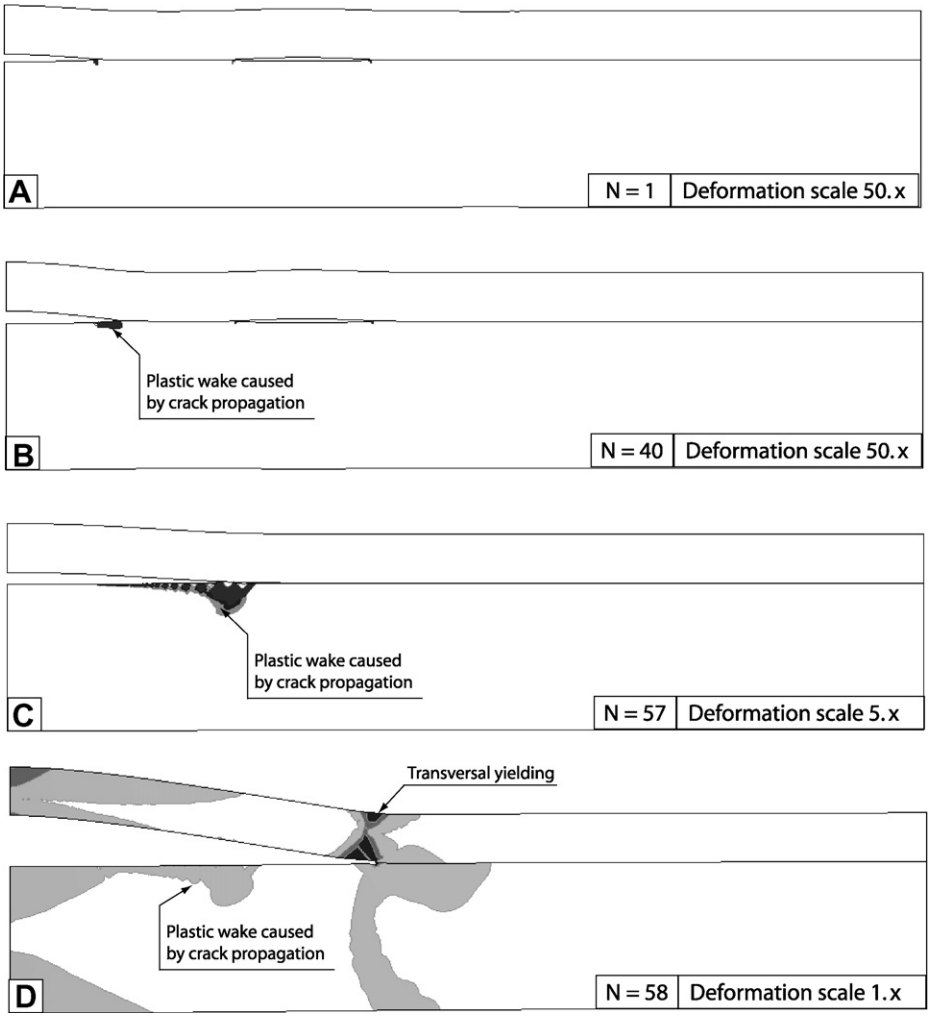

Fig. 12. Density of plastically dissipated energy in the bi-layered model, for the case $a_2 = 15$ mm: (A) after the first cycle; (B) after 40th cycle; (C) after 57th cycle (before the coalescence); (D) after 58th cycle (after the coalescence). The darker regions indicate higher levels of plastic dissipation.

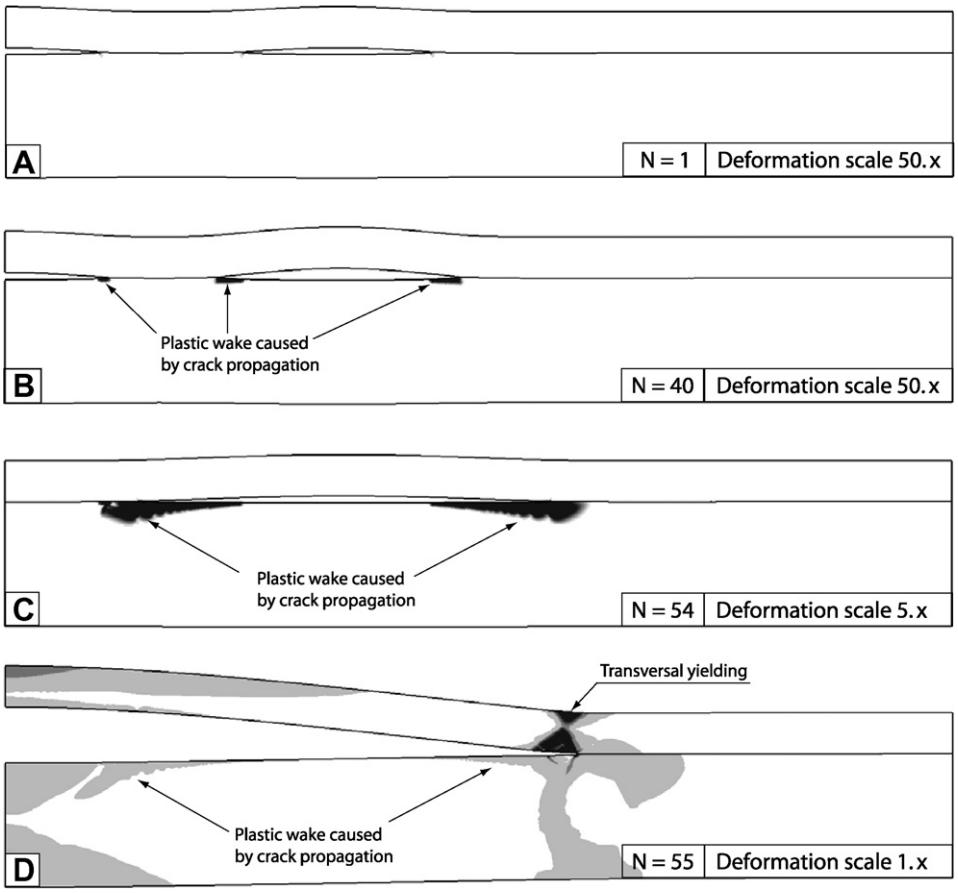

Fig. 13. Density of plastically dissipated energy in the bi-layer model, for the case $a_2 = 20$ mm: (A) after the first cycle; (B) after 40th cycle; (C) after 54th cycle (before the coalescence); (D) during the 55th cycle (after the coalescence). The darker regions indicate higher levels of plastic dissipation.

elements. The three crack-tips are labeled C_1, C_2 and C_3 as shown in Fig. 10. As in the previous sample problem, we use the plastically dissipated energy as the propagation criterion, with the critical value set to $W^p_{cr} = 0.1$ N mm for all crack-tips. For each crack-tip, the integration domain associated with the propagation criterion is defined as a discrete semi-disk ahead the crack-tip given by the domain selection function $D(x = 0; R_i = 0, R_o = 1.0, \theta = \pi/2)$ (Fig. 10). The computational time for the following examples has been estimated at approximately 24 min/computed loading cycle.

Two initial lengths of the right crack, $a_2 = 15$ mm and $a_2 = 20$ mm, with all other dimensions and material parameters kept constant, are investigated. The propagated lengths for the modeled crack-tips C_1, C_2 and C_3 are shown in Fig. 11. We will monitor the propagation of each crack-tip as a function of cycles, Fig. 11. Snapshots for selected times, showing the evolution of the cracks are provided in Figs. 12 and 13 for the two configurations, respectively.

For the case $a_2 = 15$ mm, the crack-tip C_1 of the center crack starts propagating after 13 cycles. Thereafter, the crack-tip propagates intermittently followed by accelerated crack propagation, Fig. 11A., The crack-tips C_2 and C_3 do not propagate for this case, since the crack propagation criterion is not fulfilled (i.e., $W^p(\mathbf{D}) < W^p_{cr}$). Crack coalescence occurs after 57 cycles, Fig. 12C. In the cycle that follows, the resulting crack surface is subjected to the same cyclic load (i.e., $\bar{\sigma}_{min} = 0$ MPa $\rightarrow\leftarrow \bar{\sigma}_{max} = 30$ MPa $\rightarrow \bar{\sigma}_{min} = \leftarrow$ 0 MPa), leading to very large deformation of the separated portion of the coating (Fig. 12D). The deformation is accompanied by large plastic yielding throughout the cross section of the coating (perpendicular to the coalesced crack). This may lead to spallation of the coating for a real life material.

For the case $a_2 = 20$ mm, all three crack-tips start to propagate after 11 cycles. Thereafter C_1 propagate intermittently during the entire simulation, whereas C_2 and C_3 first propagate intermittently and then in an accelerated manner, Fig. 11B. The coalescence of the cracks occurs after 54 cycles and similar cross sectional yielding is observed in the cycle after coalescence (Fig. 13) as for the case of $a_2 = 15$ mm.

In both cases, during the later cycles and before coalescence, the fastest growing crack creates a compressive stress on the nearest crack-tip of the other crack as the faster growing crack approaches the other crack. Therefore, for the higher cycles the applied pressure is not sufficient

to open the shorter crack, resulting in a decreasing crack opening displacement and insufficient plastic dissipation.

5. Concluding remarks

An object-oriented modeling frame for simulating crack propagation of cyclically loaded structures has been developed. The key feature with this modeling frame is that crack growth is initiated once a user-defined propagation criterion is satisfied. Thus, the approach does not require prior knowledge of the crack propagation rate, but the crack propagation rate is determined by the simulations. The modeling frame was developed in the context of the commercial FE code ABAQUS, utilizing Python language and ABAQUS Scripting Interface. Even though the approach was designed towards multi-layered structures, it can directly be applied to isotropic materials for simulating the crack propagation along a known path.

The modeling frame is founded on the decomposition of a two-dimensional structure into a set of continua and a set of continuum interfaces. The interfaces between two continua are described as sequences of failed and intact interface segments. The segments describing the failed portions of the interface can be assigned contact interactions. Based on the propagation criterion, the interfaces are updated on a cyclic basis, simulating the cracks propagation.

The object-oriented approach used let the user develop and implement, with very little program effort, various propagation criteria. To illustrate the possibility of utilizing user-defined propagation criteria, we implemented a new criterion based on the plastically dissipated energy.

Two benchmark problems were presented to elucidate the effectiveness of the developed modeling frame and the influence of parameters such as the critical value of the propagation criterion and load conditions. To this end, an isotropic tensile specimen with a center crack and a bi-layered system were employed. These sample problems show that, depending on the parameters used, the modeling frame is capable of capturing a stationary crack, intermittent crack growth, as well as accelerated crack growth associated with each cycle.

Acknowledgement

The authors would like to acknowledge the support from ONR-N00014-04-1-0498.

References

[1] Suresh S. Fatigue of materials. 2nd ed. Cambridge University Press; 1998.
[2] Karlsson AM, Evans AG. A numerical model for the cyclic instability of thermally grown oxides in thermal barrier systems. Acta Mater 2001;49:1793–804.
[3] Karlsson AM. On the mechanical response in a thermal barrier system due to martensitic phase transformation in the bond coat. J Eng Mater Technol 2003;125:346–52.
[4] Cojocaru D, Karlsson AM. A simple numerical method of cycle jumps for cyclically loaded structures. Int J Fatigue 2006;28(12):1677–89.
[5] Kiewel H, Aktaa J, Munz D. Application of an extrapolation method in thermocyclic failure analysis. Comput Methods Appl Mech Eng 2000;182:55–71.
[6] Oskay C, Fish J. Fatigue life prediction using 2-scale temporal asymptotic homogenization. Int J Numer Methods Eng 2004;61:329–59.
[7] Lemaitre J, Desmorat R. Engineering damage mechanics: ductile, creep, fatigue and brittle failures. Berlin: Springer-Verlag; 2005.
[8] Ingraffea AR. Computational fracture mechanics. In: Stein E, de Borst R, Hughes TJR, editors. Encyclopedia of computational mechanics. Solids and structures, vol. 2. John Wiley and Sons; 2004.
[9] de Borst R. Some recent developments in computational modelling of concrete fracture. Int J Fract 1997;86:5–36.
[10] Karihaloo BL, Xiao QZ. Modelling of stationary and growing cracks in FE framework without remeshing: a state-of-the-art review. Comput Struct 2003;81:119–29.
[11] Dolbow J, Moes N, Belytschko T. An extended finite element method for modeling crack growth with frictional contact. Comput Methods Appl Mech Eng 2001;190:6825–46.
[12] de Borst R. Numerical aspects of cohesive-zone models. Eng Fract Mech 2003;70(14):1743–57.
[13] Brocks W, Cornec A, Scheider I. Computational aspects of non-linear fracture mechanics. In: de Borst R, Mang H, editors. Comprehensive structural integrity, vol. 3. Elsevier Press; 2002.
[14] Xie D, Waas AM. Discrete cohesive zone model for mixed-mode fracture using finite element analysis. Eng Fract Mech 2006;73(13):1783–96.
[15] ABAQUS 6.5 User's Manual. 2004, Providence, RI: ABAQUS, Inc.
[16] Mackerle J. Object-oriented programming in FEM and BEM: a bibliography (1990–2003). Adv Eng Software 2004;35:325–36.
[17] Besson J, Foerch R. Large scale object-oriented finite element code design. Comput Methods Appl Mech Eng 1997;142:165–87.
[18] Commend S, Zimmermann Th. Object-oriented nonlinear finite element programming: a primer. Adv Eng Software 2001;32(8):611–28.
[19] Dubois-Pelerin Y, Pegon P. Object-oriented programming in nonlinear finite element analysis. Comput Struct 1998;67:225–41.
[20] Eyheramendy D, Zimmermann Th. Object oriented finite element programming: an interactive environment for symbolic derivations, application to an initial boundary value problem. Adv Eng Software 1996;27(1-2):3–10.
[21] Mackie RI. An object-oriented approach to fully interactive finite element software. Adv Eng Software 1998;29(2):139–49.
[22] Mackie RI. Object-oriented finite element programming-the importance of data modelling. Adv Eng Software 1999;30(9–11):775–82.
[23] Zimmermann T, Bomme P, Eyheramendy D, Vernier L, Commend S. Aspects of an object-oriented finite element environment. Comput Struct 1998;68:1–16.
[24] Schollmann M, Fulland M, Richard HA. Development of a new software for adaptive crack growth simulations in 3D structures. Eng Fract Mech 2003;70(2):249–68.
[25] Carter BJ, Wawrzynek PA, Ingraffea AR. Automated 3D crack growth simulation. Int J Numer Methods Eng 2000;47(1–3):229–53.
[26] Gullerud AS, Koppenhoefer KC, Roy A, Dodds Jr RH. WARP3D: 3D dynamic nonlinear fracture analysis of solids using parallel computers and workstations. In: Structural research series no. 607, 2004, University of Illinois at Urbana-Champaign, Department of Civil Engineering.
[27] ABAQUS 6.5 Scripting User's Manual. 2004, Providence, RI: ABAQUS, Inc.
[28] ABAQUS 6.5 Scripting Reference Manual. 2004, Providence, RI: ABAQUS, Inc.
[29] Lutz M. Programming python. Sebastopol, CA: O'Reilly; 1996.
[30] Keogh J, Giannini M. OOP demystified. McGraw-Hill Osborne Media; 2004.

[31] Evans AG, Mumm DR, Hutchinson JW, Meier GH, Pettit FS. Mechanisms controlling the durability of thermal barrier coatings. Progr Mater Sci 2001;46:505–53.
[32] Karlsson AM, Levi CG, Evans AG. A model study of displacement instabilities during cyclic oxidation. Acta Mater 2002;50: 1263–73.
[33] Bodner SR, Davidson DL, Lankford J. A description of fatigue crack growth in terms of plastic work. Eng Fract Mech 1983;17(2): 189–91.
[34] Klingbeil NW. A total dissipated energy theory of fatigue crack growth in ductile solids. Int J Fatigue 2003;25:117–28.
[35] Flores KM, Dauskardt RH. Local heating associated with crack tip plasticity in Zr–Ti–Ni–Cu–Be bulk amorphous metals. J Mater Res 1999;14:638.
[36] Bhalla KS, Zehnder AT, Han X. Thermomechanics of slow stable crack growth: closing the loop between experiments and computational modeling. Eng Fract Mech 2003;70:2439–58.
[37] Simo JC, Hughes TJR. Computational inelasticity. Springer; 1998.
[38] Solanki K, Daniewicz SR, Newman JC. Finite element analysis of plasticity-induced fatigue crack closure: an overview. Eng Fract Mech 2004;71(2):149–71.